DRAGON USER

The independent Dragon magazine

July 1988

Contents

Editorial

WHAT an irresistible opportunity — I mopped up practically everything worthy the name of news last month by going round the Ossett Show and phoning up half the Dragon world beforehand. Result: three-quarters of a page to throw around this month.

I've done two things with it: one is to reproduce a 'universal Dragon User flyer' which I hope readers with a suitable opportunity will be able to display on a noticeboard or in a window. The other is to offer 10p a line for news reporting. Don't all fall over at once — it must be new news which we haven't had from primary sources, but think about it: find two lines of publishable news and you've paid for the stamp ...

The Expert is tied up with exams for about three months, so if anyone else with a games addiction problem and a bit of know-how fancies themselves as columnist for a month, here's your chance. Send an outline if you don't want to write a column on spec, to me, at Dragonpubs.

If anyone else thinks we owe them a Bean Stalker **by the end of next week, please write. I'm still having a spot of bother collating files after the move. What I need is a reverse Hoover — one which disgorges things I thought I had dealt with just as I discover I haven't. Now, about those questionnaires ...**

Telephone number
(01) 570 8335

Editor
HELEN ARMSTRONG

Production Editor
HELEN ARMSTRONG/ARTSET

Administration/Advertising
BOB HARRIS

Publisher
DRAGON PUBLICATIONS

Subscriptions
UK £14 for 12 issues
Overseas (surface) £20 for 12 issues

ISSN 0265-177

Address: Dragon Publications, 49 Alexandra Road, Hounslow, Middlesex TW3 4HP, United Kingdom.

Published by Dragon Publications 1988

Typeset by Artset Limited, London NW1

Printed by Headley Brothers Ltd. Ashford, Kent

Registered at the Post Office as a newspaper. Dragon and its logo are trademarks of Eurohard Ltd.

How to submit articles

The quality of the material we can publish in *Dragon User* each month will, to a very great extent depend on the quality of the discoveries that you can make with your Dragon. The Dragon computer was launched on to the market with a powerful version of Basic, but with very poor documentation.

Articles which are submitted to *Dragon User* for publication should not be more than 3000 words long. All submissions should be typed. Please leave wide margins and a double space between each line. Programs should, whenever possible, be computer printed on plain white paper and be accompanied by a tape of the program.

We cannot guarantee to return every submitted article or program, so please keep a copy. If you want to have your program returned you must include a stamped addressed envelope.

Letters

This is your chance to air your views — send your tips, compliments and complaints to Letters Page, *Dragon User*, 49 Alexandra Road, Hounslow, Middx TW3 4HP.

Every month we will be shelling out a game or two, courtesy of our supplies, to the reader/s who send the most interesting or entertaining letters. So send us your hints and your opinions, send us your hi-scores and suggestions. Send us your best Dragon stories. What d'you think we are, mind readers?!

EXTRA PUFF

Old news in demand

IS it possible to give a brief summary of the contents of *Dragon User* for the past three and a half years so that I could order back copies? I am particularly interested in firmware routines. I would be exceedingly grateful.

R. Putman
26 Linten Road
Penn
Wolverhampton
WV4 4DS

Well, hoom, nobody in this office has ever done a complete index of Dragon User, for the very good reason that we never have time, but we know a man who has .... Pulser Software have published *Magbase* for Dragon User, which gives you a complete index at your fingertips, costs under a fiver, and is updated periodically. Far be it from me to interlope upon the fruits of their good work. Pulser can be contacted at 36 Foxhill, High Crompton, Shaw, Oldham OL2 7NQ.

We want articles on how to use software

MANY thanks for keeping DU going. A possible idea for a few lines might be how to use some of the more complex programs, eg Flex *Dynacalc* spreadsheet. I find that one has to spend so much time trying to get things running one runs out of available time when a few good worked examples would clear up most of what are really just silly faults.

I am not knocking programs, but it is just that instructions only become clear after you don't need them. Maybe Bob Harris has a few tips on using his BOX programs — there must be some things not covered by the idiot sheets.

J J Oakley, 9 St. Johns Road,
Clevedon, Avon, BS21 7TG

RARELY have I seen the 'documentation dilemma' stated so succinctly and with such dignity. I am in total agreement, although I will say in defence of people who write computer manuals — not that most of them deserve it — that users do occasionally come up with some very imaginative questions; you know the kind of thing — how do I use my spreadsheet to write a novel? Why can't I get my cassette out of the disc drive? And other things which reduce programmers to tears. Serve them right, too.

Bob is buried deep under a heap of paper at the moment, but he may find something to add in due course. Meanwhile, if anybody wants to write a definitive guide to using any of the major packages, please drop me a line and volunteer, stating your software and your experience. This is potentially a very useful angle, and one which DU hasn't touched on for some time.

We will pay for quality

I would like to agree with Ken Smith in the May edition when he wrote that he would be prepared to pay a higher price for some top class American software. I have been enquiring whether there was anybody at present buying software from American, and whether it was practical to do so.

Unfortunately, it appears that there is very little being produced in the USAS for the CoCo 2, which is the equivalent of our trusty Dragon. The main interest now is in the CoCO3, which has a far larger memory, and is therefore not compatible. I would love to be proved wrong on this matter.

If there is still good software I would be interested in forming a collective to identify sources and import the software. Could anyone inform me as to whether this is possible or even legal on a group basis?

Finally, if any of the companies still supporting the Dragon are considering arranging the import of such software, I would like to assure them that I am prepared to pay £10-£15 for good software.

Ian Burford
79 Hertford Road
Stevenage
SG2 8SE

WP at rest

WONDER upon wonder! This morning I receive the long awaited next issue of *Dragon User*, and what do I see but one massive article about using the printer port as a controller for external circuitry. Brilliant! Seriously if you keep this up you'll start giving people like me ideas and one day a cheap graphics card may arise.

I can see that what the Dragon has been lacking all along is a good resolution with eight colours or so, and a programmable sound generator. It shouldn't be too difficult to interface a graphics processor and a sound processor. For example, the Texas Instruments TMS 9129 NL graphics chip (which I think is incorporated in the TK 99/4A, the Einstein and the MSX machines) comes complete with its own 16K ram, so that a minimum of interfacing and external circuitry is required. The sound chips which most of the other computers (eg the Oric, Einstein) have are similar to the AY 3 8910 which provides three voices and noise with programmable envelopes etc.. I'm sure such a chip could be easily interfaced. When I think of the sort of power affordable if, say, the newest TI graphics processor (a fantastic chip, I saw it reviewed in ETI) it would provide graphics workstation capability! However, I am still learning about electronics, as it is a hobby and I'm busy enough without it!.

Thanks for clearing up the "obscure dream" in the letters page. Concerning my word processor, my Dragon is not on the blink, nor is the wordprocessor. What lack is a display! I sold my telly as I am studying and a TV would completely obliterate my chances of learning everything I need to begin research next year. Yes, Pete Gerrard was right about my being male — girls tend to have a 'typewriter' script, so they're the ones to copy lecture notes from! I hope he was correct about the doctor, in three years' time, anyway.

So until I get hold of a composite video monitor I will have to use a pen.

F R Ellahi
119 Philharmonic Court
Catharine Street
Liverpool
L8 7SD

SUCH dedication is an example to us all! How do you live without Wogan? Your informer was wrong about the 'typewriter script', by the way, unless he means 'handwriting deteriorating due to constant use of keyboard'. Anybody who has received an editorial red-ink letter will know what I mean.

Stand up and write

AS a result of placing a letter in *Dragon User* about my new magazine, *Dragon Logic*, I've had a grand total of two letters. In fact, one was from a Dragon enthusiast who didn't own a

Dragon. The Dragon world will be in a sorry state is enthusiasts contribute towards magazines but the actual users don't do a thing. You only get out of a magazine what you put into it. I know we may not all have the talents of the Cadges, D'Arcys, Gerrards, Lees etc of this world, but too many of you Dragon users are content to see your copy of *Dragon User* arrive every month, without putting an ounce of effort into it. Without input magazines don't get published. Without magazines, we're nothing.

So let's see your letters, and fast! I don't just speak for *Dragon Logic*, I'm sure many others feel the same.

Donald Morrison
Dragon Logic
72 Diriebught Road
Inverness
Scotland IV2 3QT

WELL, I bet that shook a few of you out of your socks! Spiritually, magazines depend on reader feedback.

DM has never been slow to feed back and ask questions. Dragon Logic reports that Incentive Software still supply Dragon games, although they 'left the Dragon market' two years ago. They are at 2 Minerva House, Calleva Park, Aldermaston, Berks RG7 4QW for enquiries.

64 columns with discs

THOSE of you who have a disc drive connected up to your Dragon will or may have found that Paul Harrison's 64 column screen adaptor (DU December 1987) does not work properly. This is due to the fact that when a disc drive controller is connected up to the cartridge port, the graphics pages get moved up by 1536 bytes. Therefore, page one is situated at 3072 (&H0C00), page two is situated at 4608 (&H1200) and so on up to page eight. However, all is not lost, because with the aid of a few simple pokes, the program can be made to work perfectly with a disc drive connected.

```
POKE &H7808,&H0CPOKE
&H7875,&H0CPOKE &H787E,
&H23POKE &H7888 &H0C-
POKE &H78F4,&H24
```

Also, given below is a simple Basic program that will dump the whole PMODE 4 screen to a Brother HR-5 thermal printer.

```
1 DIM A(8,1)2 FOR K=0 TO 8 :
REACH A (K,0) , A(K,1) : NEXT
3 DATA 3,3,2,1,0,0,3,1,3,3,0,0,
1,2,3,1,3,31000 ,
===============
1010 , SCREEN DUMP ROU-
TINE 1020 ,
===============
1030 PRINT£-2, CHR$(27);
"E"; 1040 PRINT£-2, CHR$
(27); "A"; CHR$(8)1050 FOR
L=255 TO 0 STEP-41060
PRINT£-2, CHR$(13); CHR$
(27); "K"; CHR$(128); CHR$
(1);1070 FOR K=0 to 1911080
T=0: S=0: FOR M=3 TO 0
STEP-1: P=PPOINT(L+M,K):
T=T*4+A(P,0): S=S*4+A(P,1):
NEXT1090 PRINT£-2M CHR$
(T); CHR(S);1100 NEXT K,L
1110 PRINT£-2, CHR$(T);
CHR$(S);1120 RETURN
```

Harvey Grey
Three Trees
Greens Farm Lane
Billericay
Essex CM11 2NY

Gemini Database

I have recently purchased (secondhand) a Dragon 32 and a copy of Gemini Marketing's *Database* program came with it. However, the previous owner had lost the instruction manual and although I can load the program, I cannot seem to find out how to return to the main menu display after entering records.

Can anyone who has this program help me with the problem? I would be very grateful if they could. Of course, a photocopy of the instructions would be even better and I would pay the postage costs.

I am also seeking a 32K expansion pack and a printer if anyone has one for sale (new or secondhand).

Many thanks.

Mark Matthews
14 Grantley Close
Ashford
Kent TN23 1UE

Seikosha shortage

COULD you or any of the Dragon User readers help me? I have a Seikosha GP100 A printer and I need a ribbon cassette for it. I have looked in all the likely shops in my area and have drawn a blank. If anyone can tell me where I can purchase the above and at which price I would be very grateful.

M. Bradley
15 Longdale Drive
Blacon
Chester
Cheshire CH1 5AT

ANY news of good deals gratefully received. Have you tried Seikosha or their representatives, by any chance? Any decent computer shop ought to be able to tell you where to get in touch. Or try Harry Whitehouse on 0636 705230.

Tandy traps with extra bytes

I read Eric Hall's description of the Tandy disc operating system with interest, but feel that some people may be misled (as I was at first) by his description of the 338 bytes in each sector. In fact the apprently arbitrary sequence of 00s, FFs, 4Es etc. at the beginning and end of each sector (described by Eric as 'system control bytes') are part of the stand double-density format (IBM System 34 format); most disc controller chips (including the Western Digital 2797 used in the Dragon) expect this sequence and strip it off, passing only the 256 data bytes to the cpu. The only time the programmer has to be aware of the extra bytes is when the disc is initially formatted.

And now a question — does anyone know a source of Tandy DOS cartridges (dead or alive, or even just a circuit diagram?) — several of us are trying to import CoCo 3s from the USA, and need a DOS cartridge to run with them, but are finding that the supplies via Tandy UK have dried up.

Bob Hall
22 Cumbria Close
Thornbury
Bristol
BS12 2YE

THANKS, by the way, to everybody who has sent in suggestions, and to those who sent in copies of information about the Dragon Professional. Even if I can't comment on each one personally, be assured they are being read and inwardly digested.

A current issue once

WHILE looking through some of my old electronics magazines, I happened to glance at the name of the editor of an ancient (March '85) edition of *Electronics Monthly*. The editor is one Helen P. Armstrong. Would this Miss Armstrong be the same as our favourite editor?

P D Smith
Univeristy Hall
Birchwood
Penylan
Cardiff
CF2 5YB

DEPENDS who your favourite editor is, does ... hang on, what do you mean, *ancient*??

Draws a correct line

I'D like to thank Bob and Ian Thomas for their DragonDOS amendments to *Stuntbiker* (or *Motorbiking* as it appears to have been re-named). However, lines 360 and 3670 were correct as printed. The statement LINE-(X,Y), where a is PSET or PRESET, will draw a line from the last point specified (or centre screen if no point previously specified) to the point (X,Y). This saves having to remember and enter extra data (just laziness on my part!)

Another point is that some may find useful is that the Dragon allows single dimension arrays of up to 11 elements (namely 0 to 10) without the arrays first being dimensioned.

Finally, thanks to whoever in the office sorted out my late February *User*. It finally arrived well after the March issue. Is this some sort of record?

Richard Boryna
71 Oakley Drive
Wellingborough
Northants
NN8 3JY

No.

News desk

If you have any new products for the Dragon — software or hardware — send us your press release

AS most of the forward news in the Dragon world was reported in the aftermath of the Ossett Show (see last month's issue of *Dragon User*), there isn't a great deal of product news to report this month. Orange Software are continuing to extend their range of disc software (Orange Software, The Garth, Star Road, Nant-y-Derry, Avergavenny, Gwent NP7 9DP). They usually have an up to date sheet with a guide to available formats on it, and can sometimes supply unlisted formats to order ... Bernd Neuner of the Siegfried Computer-Gruppe reports having seen an advertisement for Dragon 64s for $89 at California Digital, 17700 Figueroa St., Carson, California, CA 90248, USA. Write to California Digital for information and postal charges. The Siegfried Computer-Gruppe publish an A5 magazine in German and can be contacted care of Bernd at Dina-Ernstberger-Strasse 23, D-8524 Neunkirchen am Brand 1, Bundesrepublik Deutschland.

Why an ad?

Below this column you will see what looks (or should look) very much like an advertisement for *Dragon User*.

So why are we advertising in our own pages? And what has this to do with you?

Stick 'em up

The answer is very simple. Major press advertising would swallow up DU's entire budget in a couple of bytes, but we have one way of getting this advertisement into a few thousand outlets, and that is via a few thousand readers. Many community buildings, clubs and colleges have noticeboards or advertising broadsheets where special interest notices can be displayed, sometimes free, sometimes (as in the ubiquitous newsagent's window) for a few pence a week.

We are not demanding that you cut up your *Dragon User*! Most libraries have photocopiers which will make a good copy for 10p. Copy shops are more expensive, at around 12-15p for a single copy. Or you may have access to a friend or colleague's copier.

There are other things you can do, as well. Write a letter to your local newspaper or community magazine, or to another computer magazine, general interest magazine, or national newspaper (aiming for the computer club page if there is one), and simply tell them that DU exists, that it is an independent, non profit-making publication supporting the Dragon and 6809 community, and where people can find us.

Reporters, please

NOW, something in return. If you have your hands on any current news which is pertinent to the Dragon, Tandy and 6809 world, send it to the editor marked NEWSDESK, and if we use it we will pay you 10p a printed line (or £1 per ten lines) and give you a byline.

The news must be reasonably current, and must not involve any business or private interest of your own or a colleague's. Please include your source where possible. If you draw our attention to current items in other publications, so that we are able to follow the lead and verify it ourselves, we will pay half the lineage above. Payment on published lineage only, first come first served by date of postmark; we cannot use material drawn from press releases which we subsequently receive from the manufacturer (unless they arrive after we have published). Replies to readers' letters and non-Dragon-specific product information won't be accepted as news. All copy will be edited as necessary, and the editor's decision is final.

Of course, if you have a product, user group or other Dragon-related activity you want mentioned on your own behalf, we will publicise it as usual if you send a press release to the Newsdesk. Copy date is around the 15th of the month every month.

Expert's Arcade Arena

Write to 'The Expert' at Dragon User
49 Alexandra Road, Hounslow,
Middx TW3 4HP.

HELLO and goodbye. I've got to go now, to leave room for the necessary edited highlights of the *Chuckie Egg* screen designer instructions. See you in October, cheerio now. Take it away, buddy ...

Chuckie Egg designer

The program consists of three programs, **listings one, two** and **three. Listing one** recodes the original screens to a fixed length format, upon which the editor can operate, and a second program, **listing two**, is then required to convert the new format screens back to the original format, when the game is played. The new set of screens is held high in memory, above *Chuckie Egg*, and was necessary because the original screens were of random lengths making them almost impossible to edit.

To enter the machine code you must firstly type in the Basic hexloader program. You must then RUN it and use it to enter listings one, two and three (the instructions for doing this are detailed in the initial lines of the program. The hexloader, **listings one and two**, and the first part of listing three can be found in the June 1988 issue of *Dragon User*. **Listing three** continues with the full page of data 30894-32323, and the short passage 32324-32510).

The code should then be committed to tape as follows:

Listing one:
CSAVEM"1",7680,8058,0
Listing two:
CSAVEM"2",15360,15683,0
Listing three:
CSAVEM"3", 30608,32510,0

Once you have typed in and recorded these three programs, you can then compile *Chuckie Egg*+ by typing the commands below:

1) POKE113,0:EXEC 46004
2) *Ready your *Chuckie Egg* original
3) 1FORI=0TO8:READA:
POKEI+32700,A+57:NEXT:
DATA132,103,89,77,,126,-56,46,
4) RUN
5) EXEC32700
6) *Ready your cassette containing listings 1 to 3
7) CLEAR99,12345
8) CLOADM"1"
9) EXEC 7680
10) CLOADM"2"
11) CLOADM"3"
12a) CSAVEM"CH-EGG+",
13824, 32510,15657
b) CASVEM"CH-EGG+",
13824, 32600,15657
13) CLEAR9,32678:
EXEC 15420

In the above steps, two alternative commands for saving are provided at step 12, the second of which will produce an autorun saving (which can be deactivated by typing CLEAR9 before loading). Both versions should be loaded using CLOADM and the ordinary version is executed by typing EXEC.

Instructions

Upon execution the usual title screen will be displayed and the tune will be played. THere are then three extra options: E ... Enter the screen editor. (Not available if the current set of screens has been "locked"). T ... Turn the title tune on/off (no noticeable immediate response). L ... Load a complete set of screens from cassette.Upon entering the screen editor, the first screen will be displayed, and the cursor will flash in the centre of the screen. The cursor can be moved around the screen using the arrow keys. The other functions are as follows: BREAK ... Moves to the next screen CLEAR ... Exits the screen editor ENTER ... Gives access to cassette facilities and is followed by:

1 ... Save single screen only
S ... Save current set of screens, unlocked
K ... Save current set of screens, locked. Locking the screens keeps out unauthorised eyes, as they cannot be viewed or edited.
L ... Load single or set of screens. If a loading error occurs, the editor will return control to the main program.

Any other key returns. C ... Clear all the blocks in the game playing area. T ... Turns the autorail lock on/off. When the autorail is on, the current character will trail the cursor as it is moved by the cursor keys./ ... Repeat key. D ... Draw all the chickens and the farmer. These will be removed as the cursor passes over them, but their positions will remain in memory.@ ... Draw only the chickens which will appear on the screen at the start of each game.) ... Increase the number of initial chickens on the screen. (... Decrease the number of initial chickens on the screen. R ... Restore key. The screen as displayed is only in temporary storage and is permanently stored when Break, Clear or Enter is pressed. To erase the screen layout in temporary storage and restore the last permanent screen, press the R key. Chickens are not displayed on the screen layout because they are larger than the other elements of the game, and they can be placed 'on top' of other elements. The elements of the game can be placed on the screen by pressing the appropriate keys, but they must be placed according to the rules of screen design. Some of these rules are checked for as the keys are pressed (immediately), others are checked for when the user attempts to leave the screen — by pressing Break, Clear or Enter. If these conditions are not met, you will not be able to leave the screen.

Mad ducks

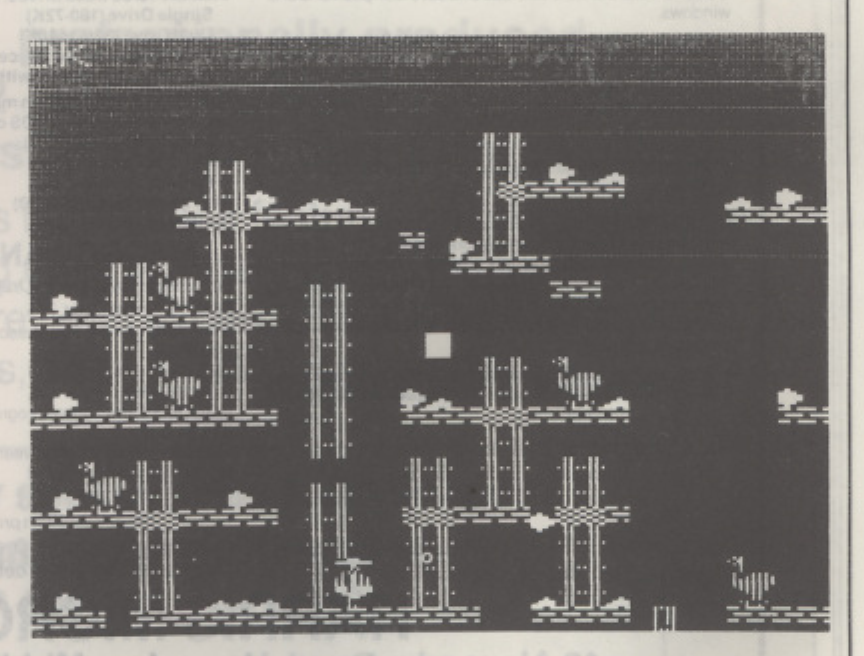
To place a wall onto the screen, press the "W" key, likewise press the "S" key for a seed grain, and the spacebar for space. These elements can be placed anywhere in the game area (the game area excludes the top three screen lines, the area taken up by the mad duck and the platform upon which the farmer begins). All screens must contain at least one seed grain, but no more than nineteen.

Ladders must always be placed as pairs of ladder blocks, since ladders are twice as wide as other elements. To place a ladder block press the "I" key; ladders (and eggs) cannot be placed at the exteme left/right of the screen, as these positions cannot be accessed by the farmer (*because of his fat tum — Expert*). Ladders cannot be placed on the bottom row of the screen as chickens would descend them and crash the program. Where a ladder intercepts (or connects with) a wall, press the "I" key to place an interception block. For the interception to be useable, the ladder must protrude by two locks in height above the wall, and the computer will check for this. Each screen must contain at least one ladder.

To place an egg onto the screen layout, press the "E" key. Each screen must contain twelve eggs.

The farmer always begins at the bottom of the screen, just left of centre. This position cannot be changed and the wall underneath the farmer cannot be removed.

To place a lift on the screen, press the "F" key when the cursor is on the left-hand side of the required lift shaft position (the lift shaft is two blocks wide — although the original author always used a width of four blocks in order to make the game more difficult). A lift shaft is indicated by an "L" symbol in



SCREEN SHOT OF CHUCKIE EGG EDITOR

the bottom left-hand corner of the shaft and this block should be replaced by another element of the game to remove the lift. Placing a lift on the screen removes any other elements in its path, and the lift cannot be placed at the edges of the screen or in the path of the mad duck's cage or the farmer's starting position. Only one lift is permitted on each screen.

Each screen contains five chickens, but not all of them start on the screen at the beginning of each game (the number can be changed, see above). Chickens are two blocks wide and two blocks high. To move a chicken to another location on the screen, press the number 1-5 referring to the chicken that you wish to replace, when the cursor is in the top, left-hand corner of the space which the chicken would take up. The positions of the chickens are stored in memory separately from the rest of the screen contents, so the chicks can be placed on top of other elements, without removing them and the chickens are only drawn temporarily (they are also not affected by the RESTORE key). The program crashes if the chicks cannot move, so they must have a wall under either foot, or ladders overlapping with their top half (that is why the chicken on level three appears to be flying!)

If entering the 2839 items of data seems too much of a daunting task for anyone, Paul Burgin says he can supply a cassette containing listings one to three for £2.50, payable to Paul Burgin, 18 Moorcroft Road, Sheffield S10 4GS (UK only, please!). (*Come on, Mr. B., we send tapes into Europe regularly and it only costs about another 50p and an extra wrap of sellotape ...*). Have fun!

(So its goodnight from him, and it's goodnight from me, goodnight!).

The Expert will be back later in the year. If anyone else wants to contribute their deeper thoughts on the finer points of arcade games in the meantime, write to the Editor.

```
30894 :082613341630A9FD80DC8A =2278
30905 :ED843088209C9D25F73516 =1A13
30916 :AE841F155E792F0A794008 =FC5
30927 :794F0979620379745277E3 =1D4A
30938 :5479170D7E220C7981437A =1582
30949 :1E447B7A2C7B5C2E7B6A40 =16E8
30960 :7B54317DF8327DF8337DF8 =262E
30971 :347DF8357DF85779874979 =2020
30982 :9D2079A14C79AC4579A853 =1E5C
30993 :79A44679CC00031A8E7C10 =13DE
31004 :BD51828E7C620D1A27038E =1297
31015 :7C53BD51827E78191F2130 =129C
31026 :88E08C6AC62204308902A0 =179F
31037 :7E78131F213088209C9D25 =15D5
31048 :F43089FD6020EE1F21301F =155B
31059 :1F109329C41FC11F26E030 =198F
31070 :882020DB1F2130011F1093 =F55
31081 :29C41F5D26CE3088E020C9 =2067
31092 :BD7C71962B4C8407972B7E =183A
31103 :77E3BD7C717E52CA86011F =1A30
31114 :21A7840D692604961C971B =12F7
31125 :BD7A2D0F697E7815860320 =1199
31136 :0D4F20E586FC20E1860520 =1B55
31147 :0A8602108C6D461022FE63 =186F
31158 :97331F209329C41F1027FE =1903
31169 :57C11F1027FE5196332OBD =1B3E
31180 :8E6D4BA680810827468C6D =1874
31191 :6626F51F209329C41FC11F =1954
31202 :2737C10B2733C10C272FC1 =1620
31213 :0D272BC10425278E6AC73A =1851
31224 :CC0707ED84BD7A2D6D80BD =2157
31235 :7A2D30881F9C9D25EF8608 =1B35
31246 :3088E0A784BD7A2D1F217E =1A07
31257 :78157E78198E6AC76F809C =2086
31268 :9D26FABD7B217E78043476 =1868
31279 :1F109329584958495849S4 =1448
31290 :5454C306001F02E6842A02 =11B3
31301 :C6048E7A5C583AEE84C608 =1DFA
31312 :A6C0A7A431A8205A26F635 =1C79
31323 :F6491D48E57A6E7A764915 =190D
31334 :4905490D491D7A7E141414 =D1B
31345 :5514141414CC33CC33CC33 =19EE
31356 :1414AA8A8A8A8A8A82AA8E =23B6
31367 :6F0ED62B308900F25A2AF9 =1EF0
31378 :108E6ACB9F25A6845F4859 =1C33
31389 :4859E7A05F48594859E7A0 =2170
31400 :5F48594859E7A0A6808403 =1D9A
31411 :A7A0109C9D25DE9E253089 =1C88
31422 :00CBA68034026DE427196A =179A
31433 :E4A601C6203D1E128E6A67 =164A
31444 :308BE6A13A1E1286FCA7A4 =236E
31455 :20E335029E25318900A786 =1868
31466 :0C3402A6A0C6203D1F01E6 =18A1
31477 :A03A30896A678605A7846A =1BAB
31488 :E426EA35029E25E68900C9 =1E27
31499 :27149E293ACC0707ED8430 =18AD
31510 :88209C9D25F78608A788E0 =244C
31521 :9E29CC0707ED890080ED89 =2085
31532 :00A0ED8900C0ED8900E0ED =285D
31543 :890082ED8900A2ED8900C2 =216D
31554 :ED8900E2ED818C6ACB25F9 =279A
31565 :CC0101FD6D53399E25A689 =1D03
31576 :00CA20209E25A68900CA27 =18CB
31587 :F06A8900CA20EA9E25A689 =2113
31598 :00CA810527E06C8900CA20 =19AE
31609 :DA860534028E6ACBBD7A2D =1CB9
31620 :6D809C9D26F735024D2736 =144D
31631 :34029E25318900BF8D441F =15BB
31642 :1058495849584954545 4C3 =19B4
31653 :06001F038E388086103402 =EE3
31664 :EC81EDC433C8206D816AE4 =2435
31675 :26F335026AE426D4350210 =1468
31686 :8E1B0CCE3600C610AEC1AF =2045
31697 :A431A8206DC15A26F47E78 =2073
31708 :19E6A0C4F84F584958491F =197E
31719 :01E6A054543A398E7C1720 =149E
31730 :0D8E7C4420088E7C262003 =EF6
31741 :8E7C3534108E7C10BD5182 =19E2
31752 :3530BD51827E7819010207 =E08
31763 :090310004C414444455253 =106F
31774 :404552524F524000454747 =107F
31785 :53404552524F5240404040 =121C
31796 :00434849434B454E534045 =1248
31807 :52524F52005345454453 40 =118A
31818 :4552524F52404040004155 =1026
31829 :544F545241494C404F4E40 =12D8
31840 :40004155544F545241494C =1318
31851 :404F464640008E6ACB4FCE =1E4D
31862 :00000F34E680C10526014C =128E
31873 :C1FC26023341C1022604D7 =15CA
31884 :342004C10327F89C9D26E3 =20E7
31895 :810C1026FF5B1183000010 =DBD
31906 :27FF4E118300131022FF46 =1511
31917 :0D3427138E6ACBA6808103 =1AE0
31928 :260DA6808102270B810327 =E80
31939 :077E7BEE810227EF9C9D25 =1E7B
```

```
31950 :E5860597339E25318900BF =18A0
31961 :BD7BDD30896A67EC848102 =1E28
31972 :260EC1022704C103260A0A =AEF
31983 :3326E7201A810327EEEC88 =1FF7
31994 :40810127EF810327EBC101 =1AB7
32005 :27E7C10327E37E7BFD8E6A =24A2
32016 :CBA6848103261AA688E081 =1F93
32027 :03271381021026FECAA688 =1F60
32038 :C08103270681021026FEBD =197A
32049 :6D809C9D26DA8E6ACB109E =214E
32060 :256FA900C90F1EA684102B =1464
32071 :0048810410250042810526 =BBE
32082 :281E129E25308900A7D61E =182F
32093 :583A1E121F1093293404C4 =121E
32104 :1FE7213504584958495849 =132B
32115 :A7A40C1E6F842016810727 =FDA
32126 :F88108260E1F10836D4710 =F7D
32137 :9E25E7A900C920E66D809C =2290
32148 :9D1026FFAA109E258E6ACB =2032
32159 :A68084034848E680C40334 =19DE
32170 :04AAE04848E680C4033404 =17D3
32181 :AAE04848E680C4033404AA =1A9E
32192 :E0A7A09C9D26D831A8250F =19E2
32203 :338E6ACBE6842A160C331F =12E4
32214 :1093293404C41FE7A03504 =171C
32225 :584958495849A7A06D809C =1F9E
32236 :9D25E09E25D633E78900CB =21E3
32247 :39D61CC031109E359E2558 =1750
32258 :308900BF3A1E121F109329 =F9B
32269 :3404C41F5858E721350444 =1316
32280 :564456C4F8E7A47E7B7ABD =28D4
32291 :7C718E7C10BD5182BD5CF2 =24A0
32302 :5052455353404C4053404B =12E6
32313 :403140FFBD800627FB8131 =1D69
32324 :277C81531027008D814B10 =11B7
32335 :27007C814C1026F98B8E1E =1A06
32346 :00BDB75BFC1E0081AB2648 =194A
32357 :5D2748C10322410F767D01 =10B4
32368 :90270CC10227040D772604 =B55
32379 :C003D7768E7000108E1E02 =1149
32390 :A6A0A7808C779026F70D76 =1EC4
32401 :1027F926BD5CF207010002 =10AF
32412 :030AFF7E50F08E1E00BDB7 =204D
32423 :5B7D1E0126B17E8B8D8E1E =1A2C
32434 :02109E25A680A7A08C1EF4 =22F3
32445 :26F77E77E3109E2530A900 =17FD
32456 :F1CCAB002015CCAB017D01 =158A
32467 :8F26065C2003CCAB038E77 =18BF
32478 :8F108E7000EEA3EDA4DF33 =2635
32489 :10BF7EFACE7EF734700F7C =2025
32500 :BD99229E33BFFFFF7E77E3 =2C0C
```

Crossword

Please get your answers in to *Dragon User Crossword Department* by the end of the month on the front cover.

The eighth Dragon Crossword emerges from the bottom of a tea chest, a little crumpled but none the worse — oh, my mistake, that's the editor! The sixth Dragon Crossword is spoken in forked Basic by G. Wright of Dunblane, who would like "Pamcomms' *Formula One*, as there's no point in asking for anything less than the best, is there?" and Ian Addisson of Chorley Wood, who would like "anything I haven't got yet, as long as it's good". The phrase is 'BASIC LANGUAGE'.

There will be a couple of free tapes from the Editor's Magic Bottomless Box for the first correct entries out of the hat each month. You can try telling us which tapes you'd like — you never know, we may have them.

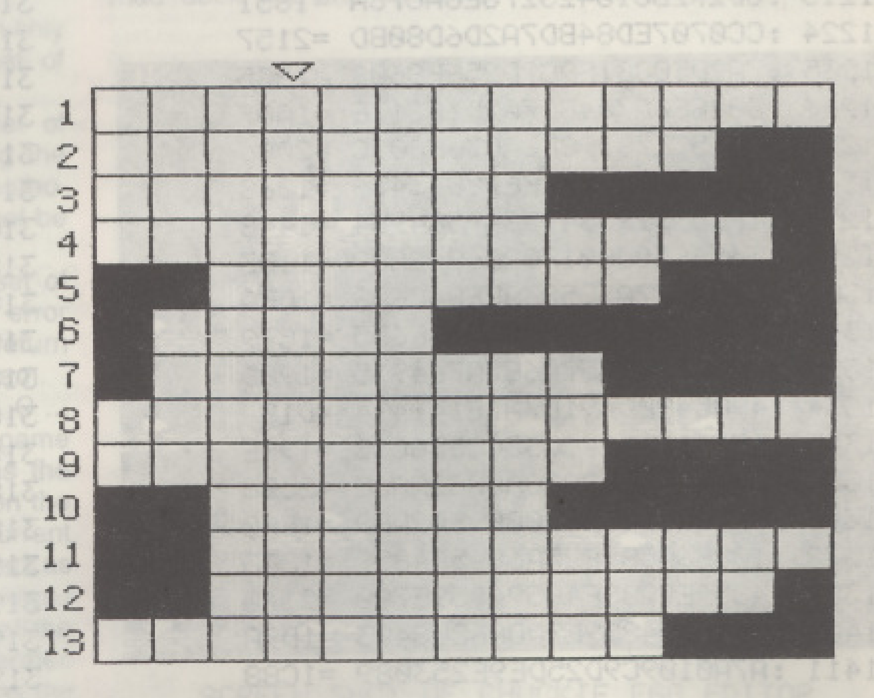
And you don't have to cut up your *Dragon User* — entries on a photostat or a plain piece of paper will do.



by Terry and Derek Probyn

All this month's answers are names of Dragon software. When the crossword is complete, the column marked with an arrow will spell out a phrase.

1. Side by side — I stop out a jinx oddly enough! (13)
2. No mice on me, with nothing on — about an alien satellite. (7,4)
3. Aliens again — play with long sax and a kiss. (8)
4 and 5. Puzzle about the celebrity from Indonesia? (7,2,3,4,4)
6. A moped ridden by Mexican Pete! (5)
7. Driven as bad as intruders. (8)
8. Subterranean chambers leading to a terrible fate! (7,2,4)
9. What the Union soldier called the Confederate. (6,3)
10. Thirst a lager cured at German POW camp. (6)
11. Lure a vulture into a trap with this! (7,4)
12. Got red rank confused, and boldly went where no man, etc. (6,4)
13. Rembrandt on TV? — His tatters lie all around him. (10)



New software for review should be sent to *Dragon User*, 49 Alexandra Road, Hounslow, Middx TW3 4HP.

By no means just your average Zap

Program: *Lucifer's Kingdom*
Supplier: Orange Software
Price: £5.99 cassette

THE picture on the cassette box is just like the title of this game — very misleading, as it shows the horned head of a demon. At first glance, this appears to be another space-type shoot-em-up game (and is even described as such on the cassette inlay) but this is very far from the whole truth. Although my two sons (and even myself when I first played it) treat it as a pure zap-em game, this should be played more like a graphic adventure game. By that I mean that you have to plan out your actions and not just necessarily shoot all the baddies.

There are other objects which can be shot, and greater scores can be achieved by destroying some of these things. The most important of these are the crystals which are placed under characters on the screen and gradually appear as the characters are shot away. They can then be collected by flying over them and a certain number are needed before you can progress from one region of six planets to the next.

One thing you cannot do, however, is to forget about the baddies in your quest for crystals, as there are a variety of types of these which come at you in a variety of ways and also fire at you differently. My pet hate are the ones that descend straight down the screen until they reach your level, and then come horizontally across at you, forcing you to retreat or move further up the screen — if you can! Others circle you Indian fashion, firing at you as they move around. Also if you are not quick enough to destroy one wave of attackers, then the next bunch swarm down upon you and they may or may not be of the same type. The choice of assailant and the number of them appears to be random, and also seems to have been generated in a way different from the usual poor Dragon randomiser.

If your ship is destroyed, then you start from that point and are not returned to the beginning, which I personally prefer. Beware, though, if you have not got past the last of the baddies which you were fighting when you were destroyed, as you will then be attacked by the same type again at the restart. Incidentally, you get five lives and I have not yet discovered a way of getting more, although you appear to have unlimited fuel and ammunition.

Only the very middle of the screen is used in this game, and not much of the rest is used for the score, etc. The movement is very smooth indeed, as is the scrolling and the speed-up poke has not been used to get the increase in momentum, just good programming. I was amazed at the response to the joystick in this game. Why isn't every game like this? Some of the baddies come at you down the very edge of the screen and these cannot be shot, no matter how far over you steer your ship. At the end of each planet, which is not very long in the early stages, the game is paused while you receive your bonus points and normally I do not like breaks in the middle of games, but believe me you need one.

It would not surprise me if this game caused some controversy in the manner of *Fire Force* about bugs, as at times it appears that you have shot straight at the enemy without harming it, but this is caused by parts of the terrain which can be shot over and over again, amassing points, but which reverse as you hit them: the first time, you can fire through them at the enemy, but not when they have turned round. Also there is a slight problem in the fire button on the left joystick, which is the one that you use in this game, will fail to work when the right joystick is connected to its own port. At times there is so much happening on the screen that you may not notice some of the enemy bullets which come in all directions.

Also some bullets appear to be directed out of nowhere, and not really from the baddies. You'll see what I mean when you play it. Another thing which throws me is that it says Player 1 in the top left hand corner of the screen with your score alongside it, and as yet I have been unable to access a two-player game as an afterthought. All in all this is a game which I really enjoy playing, and will also be enjoyed by those of you who like pure zap games. Many people will say that it is too similar to *Utopia*, but I have found that the two games need completely different tactics, and enjoy playing both. I would have liked to have given this program five Dragons, but hesitate to give the maximum to a game I have not yet completed. If the rest of the game reaches the same high standard it is certainly worth five.

Mike Stott



Find an enemy in your inventory

Program: *Underbeings of Croth*
Supplier: Dragonfire Services
Price: £4.00

THIS is an old adventure originally released by Maridan Software around 1985, and now revived by Dragonfire. Loading, as usual from cassette, is very slow and the loading screen is not very inspired.

However, when the adventure is loaded you are treated to what must be one of the best screen presentations of any adventure. The text is set out on an old piece of parchment and scrolls nicely up the screen as inputs are made. The drawback to this is that if you forget your surroundings "LOOK" has to be typed to remind you of the scenery. The instructions state that input can be in sentence form, and it took me quite a while to work this out, as you cannot "do one thing AND do another". Instead, the word THEN has to be used between the two actions.

This proved to be very useful as I played it more and more, though you have to be careful with your inputs as you have to wait for the computer to go through the two separate actions before you can make another move. Another touch I liked very much was the automatic putting of a capital to the first letter of your input and small ones to the others. This generally gave it a very professional appearance and is very pleasing to the eye. The instructions claim that 165 words are understood, and I must say that I got less "I don't understands" than usual.

As for the storyline, you play the part of an explorer trapped initially in a cage over a pit somewhere underground, and have to find treasures and then escape. There are some nice touches during the adventure, and nothing too outrageous for you to have to do in order to escape, although you are presented with some ticklish situations. A word or two of warning: take careful note of what is at each location. There is an ape-woman at one location, and I returned to this location having completed another task, and no mention was made of her. However, I did what is expected and lo and behold she was there. Be careful not to dawdle in certain locations, as I have lost my life this way several times. The story has a very outlandish setting and some very awkward problems which can, however, be solved logically. One major drawback is that the break key is not disabled, and I have completely lost the game several times and I have to admit that I'm one of those people who rarely saves their positions in an adventure game. (*That reminds me — Ed.*).

Mike Stott



New software for review should be sent to *Dragon User*, 49 Alexandra Road, Hounslow, Middx TW3 4HP.

Just one step to the real thing

Title: *Larkspur Waldorf is Trapped*
Supplier: John Penn Discount Software
Price: £3.50

THIS is the first in a trilogy of adventures with the others presumably hot on its heels (*part two out now, review shortly — Ed.*). As your mission as Larkspur in this game is to free yourself from a castle, I suppose the next title will be 'Larkspur isn't trapped any more but has landed himself in more trouble'.

The task — getting out of a castle — is not very original. The black on buff hi-res screen is divided into five sections: top left is a picture of Larkspur himself and the title, which remains throughout.

To free Mr.Waldorf you are told that you will "need a parachute. Centre top is a view of your locations, although these are limited to certain points and even then one drawing is often used for several different locations — for instance, while swimming in the lake, despite moving in different directions, you get the same view.

Top right is the inventory — not much else to say about that apart from that you start off with a hat.

Below those sections is the list of accessible directions, and then the rest of the screen is devoted to location description and your response.

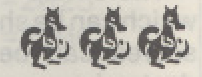
The starting point is a courtyard and you move through rooms such as kitchens, cells, an annex where there is a parachute which is typically out of reach. As per normal there are objects around such as a shield, radioactive radish and a herring which turns out to be a red one — I'm sure I've seen that somewhere else in an adventure!

Abbreviated entries such as N,E etc. are allowed, although backspacing is replaced by the clear key, which ignores the typed command.

Other standard adventure features include the maze which in this case is a complex of tunnels, and is as frustrating as normal. One thing that I thought would be frustrating was the fact that when you're fatally lacerated against an outcrop of rocks or whatever, you have to load in data from side B of the cassette. However, this is done so quickly that you hardly notice. Anyway, good players don't get killed.

All in all, I'm a bit undecided about this one, as there's nothing here that hasn't been done before. Humour is not as apparent as the title would suggest, and what there is is nowhere near as successful as say *The Quest for Life* or *The Cricklewood Incident*. Yet I've a feeling this little chap might grow on you as he progresses through his troubles, because his creators can certainly construct a good program, they just need a spark of real originality to turn it into a real adventure.

Philip Stott



Parrot goes down 3 - 0 to Dragon

Title: *Indoor Football*
Supplier: Computape (after Quickbeam)
Price: £8.95

"THE tension is mounting" here in the stadium, as the teams prepare, knowing that "at the end of the day" only one will be victorious and bearing in mind that "it only takes a second to score a goal".

Just a few football manager cliches to introduce the setting of this game — the indoor stadium. However, you don't have to wear a sheepskin coat and chunky jewellery, because you control your team by the flick of a joystick.

Before you are allowed to kick off, however, you have a few tactics to decide. Firstly, the length of the game from the ridiculously short two minutes to a sweat-inducing half hour. Then you can decide whether to play the computer or a human opponent. Also, there is a choice to the number of players in the team, from seven to eleven. Unfortunately, the opposition has to have the same number of players!

Having made these decisions, you are thrown straight into the battle of the blues-versus-reds, each displayed as teams of animated players on a sideways scrolling 3D-style pitch very similar to *International Soccer* released several years ago for the Commodore 64. To gain the ball you simply get one of your players next to it (by joystick control only), but if the opposition has possession to tackle, you have to press your well-worn red button when the ball is near. A warning, however, for certain teams: here, you can't go around kicking the opposition to bits instead of the ball, so you may have some tactical problems to overcome! Also, as there is no fouling, there's no penalties, free kicks etc. Throw-ins and goal kicks are also non-existent, as the match is indoors and the ball bounces off the boards.

To pass the ball you simply hold down the fire button (when you have possession, of course) and then let go — the strength of the kick is determined by how long you held down the button. The ball will then fly (or trickle) down the pitch to get your super striker on the ball, and to do this you once more press the button. The nearest man to the ball will start to flash, enabling you to hurtle him in a goalward direction.

Well, that's the idea. My first attempt resulted in an interval score of around 18-0 (needless to say, I had the nil). Bearing in mind another well-worn cliche that "Football is a game of two halves", I set out for the second period only to find that the opposition had lost none of their scoring ability. Full time 34-0.

Several further games against the computer resulted in similar though not quite such bad results, all with a glaring zero on my side of the score sheet. Beating human opposition is alright, but not as satisfying as trouncing the computer.

Finally, it happened. A long pass downfield straight to the feet of my attached past two defenders and a chip over the keeper. A goal that was so perfect that Peter Beardsley would give his front teeth to score it — if he had any. Perhaps he already has.

Since then, I hasten to add, I have scored more goals, several of them from kick-offs, where sometimes it seems incredibly easy to dissect the whole of the opposing team's defence.

My aftermatch report is that the graphics are good, not much sound, control is tricky to get familiar with but fundamentally simple. The game, though, is in a totallly superior league to *Crazy Foota*, although not as addictive as *Superkid*.

Meanwhile, I'm still trying to beat the computer. I'm getting there, but to finish with the most famous cliche, until that time I'm "as sick as a parrot".

Philip Stott

A goal that was so perfect that Peter Beardsley would have given his front teeth for it

New software for review should be sent to *Dragon User*, 49 Alexandra Road, Hounslow, Middx TW3 4HP.

Papa was a rollin' . . . airball?

Title: *Rola-ball*
Supplier: Preston
Price: £3.99

AN arcade-adventure called *Rola-ball* you might think could have been called *Airball 2*. That is a fair description of this program.

The game is set on a distant planet inhabited by tribes of vicious cubes and peaceful rola-balls who are in grave danger of becoming as numerous as the dodo. You have the three remaining little globes with which to save the species and to do this you guide them along numerous pathways picking up pieces of jigsaw (I've come to the conclusion that balls must have hands after all) which interlink to form a logo at the top of the screen. Get all the pieces and you live happily ever after, get jumped on three times by the cubes and you're as dead as a moa (I've used dodo already and I don't want to be repetitive).

This all sounds pretty much like *Airball* so far, but there are differences. Gone are the nails and spikes, skelton tiles, walls and thankfully you don't deflate, either. That's what there isn't. What there is is speed, these cubes don't hang about, and the pathways consist of more twists and turns and convolutions needing you to manoeuvre onto an exit disc to progress onto the next screen.

Control is totally by the keyboard, no joystick option and is quite difficult to get familiar with in such a reflex-oriented program. Up to the left is Q, down to the right is A, up to the right is P and down to the left is L. These directions are necessary because of the 3D style perspective that the game is played in. Additional keys used are the space bar to pick up pieces of the jigsaw, wnter to pause, clear to exit the screen when on a disc and finally break which forfeits a life when you appear to be stuck.

However, if you do resort to the break key, or indeed whenever you lose a life, you are not just put back on the previous screen but are jettisoned back to the starting screen which means you have to sprint through several screens you've previously laid barren to get back to where you originally were. The ironic thing about this procedure is that you often lose another life while retracing your footsteps.

The graphics are hi-res black and white without any minute detail — balls, cubes and crosses don't exactly need precision artistry, but manage to look impressive by the complexity and depth the pathways suggest. There's also a nice little tune to introduce the game but one thing that has been missed is a score feature, the only guide being the building up of the jigsaw, and this disappears at the end.

The more I play this the more I like it; at first it's a little too quick when using the unfamiliar keys. Once this is overcome through the challenge of exploring more and more alleyways requiring greater ingenuity and speed drives you on. There are forty-nine pieces of jigsaw to collect, with only three lives, so the odds are a bit stacked against you, although you can exit screens without getting all the pieces, so you can play your tactics and draw your maps in advance of completion.

This game has obviously been influenced by *Airball* and is still very well written, but the moral is that when you look like your father it's hard to follow in his footsteps with your own talents. I'm looking forward to the grandson, though!

Philip Stott



All the answers, but where do you look?

Title: *Everything you always wanted to know about OS-9.*
Supplier/author: Jason Shouler, BCS, 70 Victoria Road, Parkstone, Poole BH12 3AE.
Price: £18.95

BEING a late comer to the pains and pleasures of OS-9, I must admit I have been finding it hard going. One possible solution could be Jason Shouler's book *Everything you always wanted to know about OS-9 but didn't know where to look.* Priced at £18.95 it seems a bit expensive, but it does include a disc containing OS-9 utilities.

It has been my experience in the past that books, films, etc. with titles like this rarely live up to their titles. This book is no exception — regular readers of the American magazine *Rainbow* will recognise the contents as KISSable OS-9, a regular monthly feature in the magazine. The KISS is an acronym for Keep It Simple Stupid. The series was meant to help Tandy CoCo users get to grips with what was then a new operating system. Jason's book is in fact a compilation of these articles reprinted, bound in a plastic clip and with a cartridge paper cover.

The book's biggest failing is that it is not indexed, so it is necessary to read the whole book through to find out the answer to what might be a relatively simple problem. For what is obviously supposed to be a reference work, this is a really serious fault (*hear, hear — Ed.*). To make matters worse, as with our beloved *Dragon User, Rainbow* was subject to the occasional printers' error. These errors are usually corrected a few articles later. In a serious republication of a series of articles, the errors would have been edited out. As it is, you could spend hours trying to make one of the projects work, only to find that the information is incorrect and is corrected four or five pages later.

A prime example of this is the part that advises how to run forty track drives on a Tandy OS-9 system (the standard for Tandy is 35). We are told that this is a fairly simple excercise; just change the data held at two addresses. Unfortunately, the addresses given are not right, and are corrected three pages later.

One of the more pleasant things about magazine articles is that the writer answers readers' queries. When struggling to learn a new system it is seasy to get the impression that only you are having these problems and perhaps it is really beyond you. It is comforting to know that others have problems with the system. Anyone who has read the manual supplied with the original system disc from Dragon Data will appreciate the way the articles are written. They are in a language that even I can understand (well, almost) and presented in a way that is both well explained and quite readable. The real crime here is that there really is a wealth of information contained in this book, if you are prepared to searching for it, with sections devoted to Basic09, Pascal and the use of the C complier all of which are available to the OS-9 user. However, therw is no real continuity and you might find a large gap between one Basic09 article and the next.

To summarise, this could be a very useful book, spoiled by the lack of any editing or an index. At £19.95 it does seem somewhat overpriced. However, the size of the market must be taken into account. Four years ago it would have been edited, indexed and printed by the thousand for half the price. How times change: these days Jason will be lucky to sell a couple of hundred and will probably only just cover his production costs.

You may have noitced that so far I have made little mention of the utilities disc. This is because so far I have found no details of what they do or how to use them. By now you will have guessed that what I really need is *OS-9 for absolute beginners.*

Ken Smith



Memory and assembly

Pam D'Arcy maps assembler tools into the Dragon's memory

```
FIGURE 1      MEMORY USAGE WHEN ASSEMBLING      YOUR FIGURES
$0000         System workspace
              (A)     ($0400 bytes)

$0400         Text screen
              (B)     ($0200 bytes)

              DragonDOS type disk
              system workspace
              (C)     ($0600 bytes)

$0600         Graphics screen pages
              (default at switch on
              (= 4 pages @ $0600 bytes)
              (D)     ($1800 bytes)

$1E00         Basic program area
              (actually starts at $1E01)
              (E)

              Basic program arrays/
              variables area
              (grows as required to
              higher addressed memory)
              (F)


$7F36         System hardware stack
              (grows as required to
              lower addressed memory)
              (G)

$7F37         Basic string data stack
              (length=first value of
              (last CLEAR statement)
              (H)(default $00C8 bytes)

$7FFF         Reserved machine code area
              (length=second value of
              (last CLEAR statement)
              (I)

              DeltaDOS disk system
              workspace
              (J)     ($0600 bytes)

$8000         Basic interpreter ROM
              (K)

$C000         Cartridge area (ROM)
              (L)

$E000         Possibly unused area
              beyond end of cartridge
              software (ROM)
              (M)

$FF00-        System workspace
$FFFF         (N)
```

WHEN at this stage a reader has the nerve to admit that he still hasn't managed to assemble anything even though using the assembler software in the machine code articles, one cannot help but feel that he cannot be alone and something should be done about it! A few other queries have arisen concerning memory, so it seems right to give it another go.

Unmodified Dragons have 65536 individual accessible memory locations =bytes uniquely identified by referring to them as having memory addresses (like postal addresses) 0 through to 65535. On a Dragon 32, apart from 256 bytes at addresses 65280-65535, the top half of memory (addresses 32768-65279) can be looked at = read = PEEKed but, rather like removing the piece of plastic from the spine of a cassette to prevent it being overwritten, it is made of material that cannot be overwritten (=ROM, Read Only Memory). Corruptable memory is known as RAM (Random Access Memory). Switching on a Dragon 64 gives an identical memory map to the 32, including the ROM. There is further comment on the 64 at the end of these jottings.

All into memory

I am only really familiar with one assembler (Dream) so some of the following comments may not be applicable to your software. However, generally, there are two phases: Entering/editing text (source code) as in the nature of editing a Basic program. There will usually be a SAVE/LOAD text facility, and assembly, ie converting the text (=source code) to machine language (=machine code=binary code). There may be a save binary code facility built in to your assembler or, if using an assembler like Dream, you quit the program and save it like any other piece of machine/binary code (CSAVEM or equivalent command).

To achieve an assembly, one needs to be able to fit into memory the assembler program and its workspace, the source code (=text), and the resulting machine code (=object/binary code)

The assembler workspace will include what is know as a symbols table or list that is created while the source code is being converted to object code. The symbols list contains all the names that appear in the 'label column' of the text and the memory address/value that the assembler has worked out (=assigned) for it that are subsequently used in the resulting object code.

All these components of assembling need to be in separate areas of memory — if they start spilling into each other, corruption will occur, either because the assembler program itself has been over-

written or because overwriting of the symbols table/text/object code while assembling will give indeterminate results and almost certainly incorrect object code. These areas also need to be kept clear of system workspace, including disc workspace, to avoid catastrophes such as disc corruption caused of involuntarily activating drives.

Figure one shows a memory map split into sections lettered A-N for reference. Jot down your own figures for your system on the right hand side of the chart. Note that sections (C) and (J) are only present if a disc cartridge is attached and only one of them is then relevant depending on whether the DOS software being used is DeltaDOS (J) or other — DragonDOS, Cumana, SuperDOS all utilising the same workspace area (C). Both types of system use the same length of workspace ($600 bytes) and addresses of areas preceding/following on from them respectively are suitably adjusted.

The documentation with your assembler should give an indication of where in memory text (source code), object code, and possibly its workspace (including the symbols table) are stored. With some assemblers, the text (source code) is made to appear as a Basic program or source code appears alongside Basic statements. If you are using an assembler written in Basic, it may well be that generated object code is stored in the graphics page's memory or one is asked to issue an appropriate CLEAR statement before commencing (or one may be included within the assembler program itself).

Dream's reply

The following is with specific reference to *Dream* and in answer to queries arising from its use in previous articles.

All versions of *Dream* are machine code programs supplied with documentation that includes a memory map for the variant purchased. If the Dream cartridge is being used, the cartridge is plugged while the machine is switched off and does not autorun like a games cartridge. Dream is not activated until the appropriate EXEC has been typed in. Before using the cartridge or loading in a cassette or disc version of Dream, a CLEAR command needs to be typed in to reserve memory for the Dream program itself (if not on cartridge,) its workspace and text/symbols table/object code area. Cassette and disc versions of the Dream program are loaded into the highest free area of memory that it can be accommodated in (avoiding DeltaDOS workspace for such systems).

I didn't realise that I had been inconsistent in my articles in that some mention 'after CLEAR 200,20000' whereas others are 'after CLEAR200,&H5000'. The amount of memory cleared has nothing to do with the version of *Dream* being used but is the amount of workspace reserved. I was using round figures and tend to use '20000' (which is $4E20) if typing in a memory value in decimal and &H5000 (20480) if using hex. I do tend to use hex more often as assemblers are hex based. (I must say that £20 spent some years ago on a calculator with decimal/hex displays was money well spent; they are still readily available at that sort of price — otherwise for hex/decimal conversion you can always type in the likes of ?HEX$(decimal number) or ?&Hnnnn on the Dragon keyboard).

To take, then, the example CLEAR200,&H5000 immediately after switching on (no disc drives) leaves memory as in **figure two**.

```
FIGURE 2  AFTER CLEAR200,&H5000

$4F37  System hardware stack (G)

$4F38  Basic string data (H)

$5000  Reserved machine code area (I)

       (no DeltaDOS attached) (J)

$8000   Basic interpreter (K)
```

```
FIGURE 3  DREAM WORKSPACE ANALYSIS

$5000  Not used

$5001   Object code
       (grows as required to
       higher addressed memory)

        Symbols Table
        (Starts at lowest address
        end of text area and grows as
        required to lower addressed memory)

$5BFF   Text (source code) statements
        (Starts at address $5BFF
        and grows as required to
        lower addressed memory)

$5C00   Other reserved Dream workspace
        (always $200 bytes long)

$5E00   DSKDREAM program

END

(C) PAM D'ARCY
28MAR88
```

An advantage of the *Alldream* cartridge is that all the space in section (I) ($5000-$7FFF — 12288 bytes) is available for text/object/symbols table and *Dream* space. *Dream* on cassette or disc otherwise depletes this available space. Using, say, DSKDREAM, the program is loaded at address $5E00, meaning that $5000--$5DFF (3584 bytes) are available for text etc. *Dream* uses the workspace dynamically — that is, it doesn't affix artificial limits to the size of each of the text/symbols/object categories of memory. **Figure three** is a diagram of the machine code area (I) when using DSKDREAM.

Continuing with *Dream* as the example, on first loading up, type EXEC and reply N to 'text in memory' prompt. Although I am a great advocate of regularly saving source code to cassette/disc, it is very useful to be able to flip in and out of *Dream* retaining existing source code in memory (Y to prompt) rather than to have to reload a saved file afresh each time. On entering *Dream* for the first time (or after saying N in order to clear old source from memory), you are presented with a blank screen. This is the 'edit source' mode where you type in and edit source code statements (without line numbers). This starts filling the workspace

memory, growing down in memory addressing from the original highest available memory address (EXEC address--$201=$5BFF in this example). The symbols and object areas only come into play when assembly is requested - that is, when pressing Break/A/enter to assemble the source code currently contained in the text area.

Should this workspace become filled, Dream gives the message FULL, allowing you to quit the program and reserve a larger workspace, say, CLEAR200, &H4000 (returning to Dream with Y for text in memory).

Save the current source text then type break/A/enter to asssemble the program. As it assembles the source, the symbols table will be compiled growing down in memory addressing from the end of the text area then the generated object (machine) code is put into memory starting at the address+1 of the second value of the last CLEAR statement (eg $5001 if the last CLEAR was 200,&H5000) — an exception being if the PUT directive has told the *Dream* to put it elsewhere in memory.

Assuming that the code has been assembled to address $5001 and the assemble is error free, press break/Q/enter to return to the source (edit) screen then press break/Q/enter again to quit Dream. Now the generated machine code can be saved to cassette/disc (CSAVEM filename, &H5001,end address,exec address or equivalent command.)

This article is only intended to be jottings, so as it is already larger than expected, I cannot add any further real detail. However, if you start running short of space, progressively reduce the CLEAR 200,address figure. If OM occurs, reduce the number of graphics pages (one can achieve PCLEAR0 by a few judicious POKEs as mentioned in *Dragon User* from time to time). The 200 in the CLEAR statement could be reduced but I do not recommend reducing it to much below 100. DSKDREAM users have the lovely facility to split the source code into sections that can be called from disc for assembling a program whose source code is larger than could otherwise be accommodated on a Dragon32. Without DSKDREAM, it is up to the user to arrange source code into separately assembled, linkable modules.

The additional RAM can be used with *Dream*. I have never enquired as to whether *Dream* is relocatable and suspect that it is as far as load position is concerned but it probably uses 32K mode ROM calls so cannot be used in 64K mode. As I use discs, which also excludes 64K mode, I use the technique shown in *Dragon Answers* from time to time where RAM memory is mapped in, copying the contents of the 32K Basic ROM and cartridge (DragonDOS/Alldream) area to the identical addresses in RAM. This still leaves $E000-$FEFF (7936 bytes) free for PUTting object code into, leaving more lower memory space for source code text. I write programs in relocatable (position independent) code so that they can run unchanged at a lower address and by using the 32K mode Basic ROM routines have Dragon32 and Dragon64 (32K mode) compatibility.

Winners and Losers

Every month Gordon Lee will look at some prize programming

IN the mail this month comes a letter from Jim Finley of Romford:

"I'm knocking on for 67 and had never touched a computer twelve months ago. My son then gave me a Dragon 32 he was discarding (*shame on you, Finlay Jr.!!*), with a couple of text books and some games to play. I soon wanted to do more than play games and started on a text book and rapidly got bored because its instructions seemed to lead only to trivial output, and it had errors anyway. It simply didn't have guidance on how to do the things I wanted to do. Turning to the official Dragon manual left me even more confused.

"Another textbook seemed a splendid thing; it gave a series of modules, explaining how they worked. There was only one thing wrong with them — they didn't work! Even I managed to spot some of the bugs but had finally to give up on the things. It was at this stage that I found that there was a magazine called *Dragon User* and promptly became a subscriber."

Jim then makes some comlimentary remarks about the mag and goes on to say that he has leaned more about programming from DU than from any other source. This has enabled him to try some of the competition problems to put his programming to the test. (His first attempt was at the November puzzle — probably the most tricky puzzler set in recent months.) However, with enthusiasm undimmed, Jim has since tackled the more recent problems and says that he looks forward to future challenges.

The difficulty of learning programming (*Or anything else — Ed.*) from books alone has been a constant feature of readers' letters asking for advice, and I have given hints and tips in past issues. Clearly, the ability to program even simple routines with confidence is an advantage, not only in respect of the competition, but for other applications for which commerical software is not available. Listings from books are all very well, provided that they do the task you require, and that they are themselves bug-free.

Books on programming I would recommend to the beginner are:

Easy programming for the Dragon 32 and *Further Programming for the Dragon 32* both by Ian Stewart and Robin Jones (Shiva Publishing), and *Programming the Dragon 32* by Peter Lafferty (Newnes Microcomputer Books).

The first two of these, which are best used as a pair, explain and enlarge upon most of the commands outlined in the manual. Each is illustrated with a short routine showing the command in action, most of the listings being under a dozen clearly printed program lines. Unfortunately, these may now be out of print and only be available second hand.

However, if I were to have to choose just one book (with which to be marooned on a desert island) then my choice would be Peter Lafferty's book. This provides a broad outline of Dragon Basic, and is obviously written by someone well versed in the little idiosyncracies of the Dragon (and which frequently trip up unsuspecting competitors to the competition questions!)

On the subject of programming generally, the important thing is to be thoroughly familiar with each of the commands, and exactly what each one does, and the result obtained. Once understood in isolation, the effect of combining these commands to build program lines, and from there routines, can be readily appreciated. To give an analogy: if a photographer understands the effect of altering the shutter speed on the camera, and the effect of adjusting the aperture, he can combine the two to give him the result that he requires.

Of course, what no book can tell you is how to translate the task to be performed into the program. This is just a matter of experience in which the Basic commands form a 'toolkit', and it is up to the programmer to select the right tools for the job. The knack of 'seeing' how a problem can be adapted into a working listing is the important bridge between problem and program. The simpler the routine, the clearer the effect becomes. For example, can you devise a short routine which will exchange the values held in two variables? In other words, given two variables X and Y, place the current value of X in Y and Y in X. (These variables are assumed to hold a different value.) If you are unable to see the solution at once you will probably end up with the same value in *both* variables, the second value having been lost. However, if the same problem is presented graphically with everyday objects, the solution becomes glaringly obvious: suppose I have a red glass containing white wine, and a white glass containing red wine. The task is to switch the wine so that the colour matches the glass. Clearly a third glass (variable) is needed to save the contents of one of the glasses while the switch is being made. Call this variable Z, and the routine becomes:

```
Z=X: X=Y: Y=Z
```

Little need be said this month about the January competition, which, judging by the number of entries, provided little difficulty. The only 'catch' is likely to be the 'ghost' character formed by STR$ command, (see *The Answer, Dragon User* April 1988, p. 23).

Data grows on trees

Jonathan Cartwright spells out data storage on the Dragon 32

THE main topic I intend to investigate is that of trees. Trees normally mean oak, pine and Christmas, but in computing a tree is a way of storing data. Basic, as such, cannot handle trees, although Pascal can. So what is the use of writing about it? Well, it's a useful technique that can be "fudged" through Basic, and in any case it won't hurt you to know the theory behind data storage.

If you would care to cast your eyes over **figure one**, you should get the gist of what a binary tree is. Basically, it is a very efficient way of storing and searching data. When the numbers are put in the tree, they are organised so that:

1) Numbers greater than that in the current node are put to the right hand side.

2) Numbers lower than that in the current node are put to the left hand side.

If this is a little confusing, then take a look at **figure two**.

Now we come to the important idea of search through the tree. Because all the values higher than that of the parent node are to the right, and all the values lower than the parent node are to the left, then we can instantly rule out a large chunk of values. If, using the example in **figure two** we are searching for the value 20, then we can instantly rule out all the values less than 20, ie all those to the left. We can continue doing this at every node until we either find the number or decide that it is not in the tree at all. This is a distinct advantage over normal searching procedures, where we would go through every value until we either ran out of values or found the value that we were looking for. Using the tree on small amounts of data you will not notice any great increases in speed when searching. However, when I originally wrote this program on a mainframe, I was able to enter vast amounts of data. When searching such large amounts of data, you will notice a great improvement.

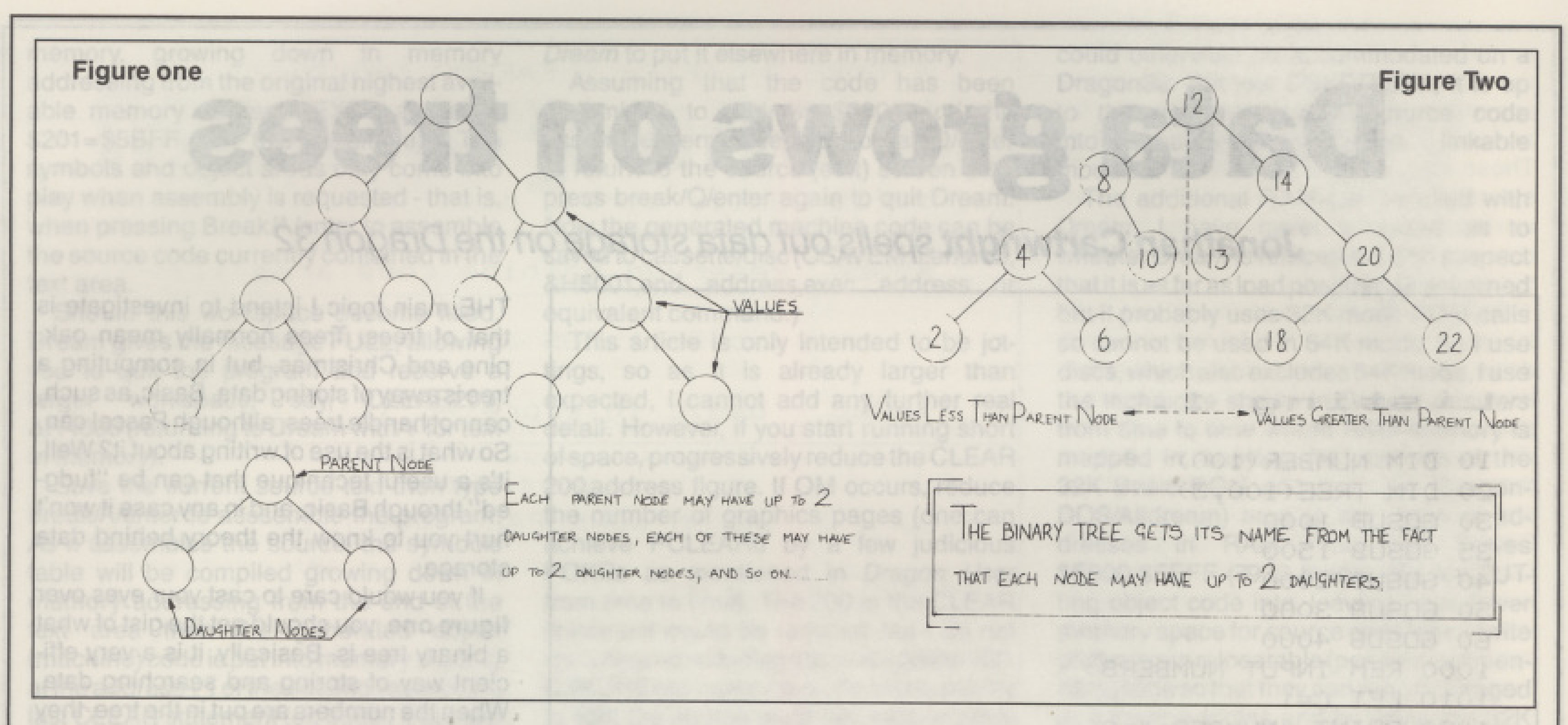
At this point it would seem appropriate to give you a program of some sort. **Listing one** is that program.

The program simulates a binary tree in Basic. It will allow you to enter up to 100 numbers into the tree, and subsequently search it. I have 'grown' the tree as a two-dimensional array' TREE(100,3). The 100 is the amount of numbers that you can put

Listing 1.

```
10 DIM NUMBER(100)
20 DIM TREE(100,3)
30 GOSUB 1000
35 GOSUB 1500
40 GOSUB 2000
50 GOSUB 3000
60 GOSUB 4000
1000 REM INPUT NUMBERS
1010 LET C=1
1020 PRINT "NUMBER:";C
1030 INPUT "ENTER VALUE (-999 TO END):";NUMBER(C)
1040 PRINT
1050 IF NUMBER(C)=-999 THEN C=C-1:RETURN
1060 LET C=C+1
1070 GOTO 1020
1500 REM CREATE TREE
1510 LET TREE(1,1)=NUMBER(1)
1520 FOR LOOP=2 TO C
1530 LET V=NUMBER(LOOP)
1540 LET C2=1
1550 LET V2=NUMBER(C2)
1560 IF V<V2 THEN GOTO 1620
1570 IF V=V2 THEN GOTO 1670
1580 REM GREATER THAN
1590 IF TREE(C2,3)<>0 THEN C2=TREE(C2,3):GOTO 1550
1600 IF TREE(C2,3)=0 THEN TREE(C2,3)=LOOP:C2=C2+1
1610 TREE(LOOP,1)=V:GOTO 1670
1620 REM LESS THAN
1630 IF TREE(C2,2)<>0 THEN C2=TREE(C2,2):GOTO 1550
1640 IF TREE(C2,2)=0 THEN TREE(C2,2)=LOOP:C2=C2+1
1650 TREE(LOOP,1)=V:GOTO 1670
1660 REM EQUAL TO
1670 NEXT LOOP
1680 RETURN
2000 REM SEARCH
2010 INPUT "VALUE TO BE SEARCHED:";V
2020 LET C2=1
2030 LET V2=TREE(C2,1)
2040 IF V<V2 THEN GOTO 2090
2050 IF V=V2 THEN GOTO 2120
2060 REM GREATER THAN
2070 IF TREE(C2,3)<>0 THEN C2=TREE(C2,3):GOTO 2030
2080 IF TREE(C2,3)=0 THEN GOTO 2150
2090 REM LESS THAN
2100 IF TREE(C2,2)<>0 THEN C2=TREE(C2,2):GOTO 2030
2110 IF TREE(C2,2)=0 THEN GOTO 2150
2120 REM EQUAL TO
2130 PRINT"VALUE ";V;" FOUND."
2140 GOTO 2160
2150 PRINT"VALUE ";V;" NOT FOUND."
2160 PRINT
2170 INPUT"SEARCH AGAIN (Y/N)";A$
2180 IF A$<>"Y" AND A$<>"y" AND A$<>"N" AND A$<>"n"
 THEN GOTO 2170
2190 IF A$="Y" OR A$="y" THEN GOTO 2010
2200 RETURN
3000 REM DISPLAY TREE
```

```
3010 FOR N=1 TO C
3020 FOR V=1 TO 3
3030 PRINT TREE(N,V);
3040 NEXT V
3050 PRINT
3060 NEXT N
3070 RETURN
4000 REM END
4010 END
```



List of numbers in tree

12 The first value is always the parent node
8
10
14
20
13 The other values
4 are daughter nodes
6
18
22
2

Figure three

Element no	Value	Less than pointer	Greater than pointer
1	12	2	4
2	8	7	3
3	10	-	-
4	14	6	5
5	20	9	10
6	13	-	-
7	4	11	8
8	6	-	-
9	18	-	-
10	22	-	-
11	2	-	-

The pointers refer to the element number

Figure four

The number 13 is to be added to the tree. As it is greater than the parent node 12, we know that it must go on the right hand side. The next node to the right is 14. Our value, 13, is less than this, so it must go to the left. There is no node to the left, and so we create one with our value in it.

in the tree. The 3 takes a little more explanation. The first value is the number itself. The second is the less-than pointer and the third is the greater-than pointer. "So what have these three pointers to do with anything?" I hear you cry. Well, you know that values less than the current value go to the left in the tree, but the Dragon doesn't. You must tell it, and this is what the pointer is for.

The pointer tells the Dragon that the next value less than the current one is in a particular place in the array. The greater-than pointer works in the same way. Again this may be confusing, so for all those completely stumped by pointers refer to **figure three**.

You now know how to search a binary tree, so how do you create one to begin with? This is very similar to searching. The only difference is that if the required value isn't in the tree then you tag it onto the end, taking into account whether it should go to the left or right. **Figure four** is prescribed for those in trouble.

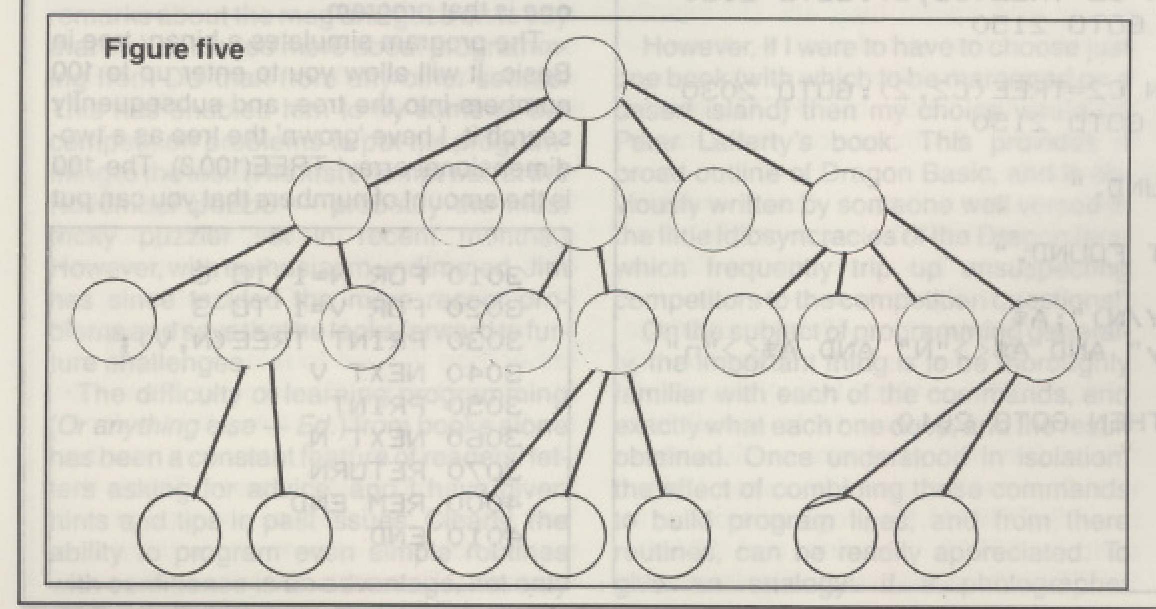
Now you can type in the program listing and use it. In this form it isn't a particularly useful program. However, it demonstrates the principle of trees and it may come in useful.

To demonstrate how trees can be used on a 32K Dragon, I'm going to give you a more useful program which is, in fact, a spelling checker.

If I wish to store a large number of words in the confines of my Dragon 32, I would be rather stupid if I tried to store each word in a separate string. Why? Well, strings take up a lot of memory and are inefficient if there are lots of small ones. So how am I going to do it? Well, I'm going to use an n'ary tree. What is an n'ary tree...?

Each node of a binary tree may have up to two daughters. In an n'ary tree, each node can have up to n daughters, where n is any number. Glance at **figure five** at this point.

From now on it gets a little harder. For simplicity, I am going to try to show you how the word ABDC would be stored in an n'ary tree. The 'parent node' is going to be A. It is going to have a daughter B. B has a daughter D, who has a daughter C. Thus the four letters are linked together. **Figure six** should make things clearer. We now know how the letters in each word are linked together. What if we add another word to our 'dictionary'? Let's take the word BBAB.



Listing 2

```
1 REM SPELL CHECKER
2 CLEAR15000
3 MAXLENGTH=20
4 DIM D$(28,MAXLENGTH-1)
5 GOSUB 10
6 GOSUB 17
7 GOSUB 24
8 GOSUB 61
9 STOP
10 REM *****CLEAR ARRAY
11 FOR I=1 TO MAXLENGTH-1
12 FOR J=1 TO 28
13 LET D$(J,I)=STRING$(28,48)
14 NEXT J
15 NEXT I
16 RETURN
17 REM *****ENTER WORDS
18 INPUT "ENTER WORD TO BE ADDED TO THE    DICTIONARY:";W$
19 GOSUB 31
20 INPUT "ANY MORE WORDS (Y/N):";A$
21 IF A$<>"Y" AND A$<>"N" THEN 20
22 IF A$="Y" THEN 17
23 RETURN
24 REM *****SEARCH WORDS
25 INPUT "ENTER WORD TO SEARCH:";W$
26 GOSUB 43
27 INPUT "ANY MORE WORDS (Y/N):";A$
28 IF A$<>"Y" AND A$<>"N" THEN 27
29 IF A$="Y" THEN 24
30 RETURN
31 REM *****ADD TO DICTIONARY
32 L=LEN(W$)
33 IF L<2 THEN PRINT "word too short!!!":RETURN
34 IF L>MAXLENGTH THEN PRINT "word too long!!!":RETURN
35 FOR I=1 TO L-1
36 X=ASC(MID$(W$,I,1))-63
37 Y=ASC(MID$(W$,I+1,1))-63
38 IF I=L-1 THEN F$="2" ELSE F$="1"
39 IF MID$(D$(Y,I),X,1)="2" THEN F$="2"
40 MID$(D$(Y,I),X,1)=F$
41 NEXT I
42 RETURN
43 REM ****SEARCH DICTIONARY
44 L=LEN(W$)
45 IF L<2 THEN PRINT "word too short!!!":RETURN
46 IF L>MAXLENGTH THEN PRINT "word too long!!!":RETURN
47 E=0
48 FOR I=1 TO L-1
49 X=ASC(MID$(W$,I,1))-63
50 Y=ASC(MID$(W$,I+1,1))-63
51 IF I=L-1 AND MID$(D$(Y,I),X,1)<>"2" THEN E=1
52 IF MID$(D$(Y,I),X,1)="0" THEN E=1
53 NEXT I
54 IF E=0 THEN PRINT "word spelt correctly.":GOTO 60
55 PRINT "word not found!!!"
56 INPUT "DO YOU WISH TO ADD THIS WORD TO THE DICTIONARY
   (Y/N):";A$
57 IF A$<>"Y" AND A$<>"N" THEN 56
58 IF A$="N" THEN 60
59 GOSUB 31
60 RETURN
61 REM *****PRINT 'PAGES'
62 FOR I=1 TO MAXLENGTH-1
63 PRINT "PAGE ";I
64 PRINT CHR$(96);:FOR C=1 TO 28:PRINT CHR$(95+C);
   :NEXTC:PRINT
65 FOR J=1 TO 28
66 PRINT CHR$(95+J);D$(J,I)
67 NEXT J
68 PRINT ":"
69 EXEC41194:NEXT I
70 RETURN
```

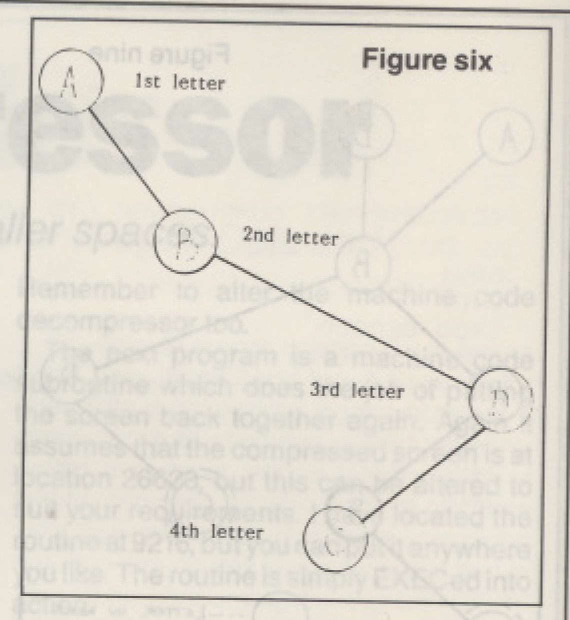


Figure six

Figure seven shows this, along with abdc. This is getting more complex. Before we try to write a piece of software, I'm going to drop you in at the deep end and add the word BBABA. This can be seen in **figure eight**. It may look as though BBABA is a valid word while BBAB is not. This problem can be avoided by having a 'terminator' in a node to signify that the word can end here or carry on. In addition to this we also need a 'special terminator' to signify that the word MUST end here. Look at **figure nine** for an example. Now you can see that things are much more clear.

You may or may not have noticed that using this system of storage you can 'fool' the dictionary into thinking that a word exists when it doesn't. For example, looking at **figure nine** you could be forgiven for thinking that the word ABAB exists. This is one of the drawbacks with this kind of data storage. However, when using real words it does not present too much of a problem.

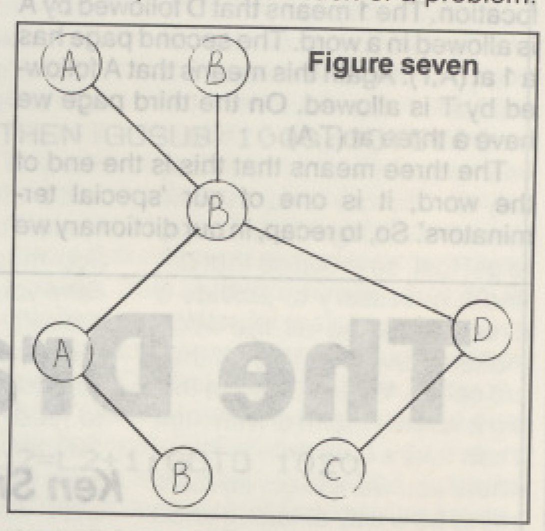


Figure seven

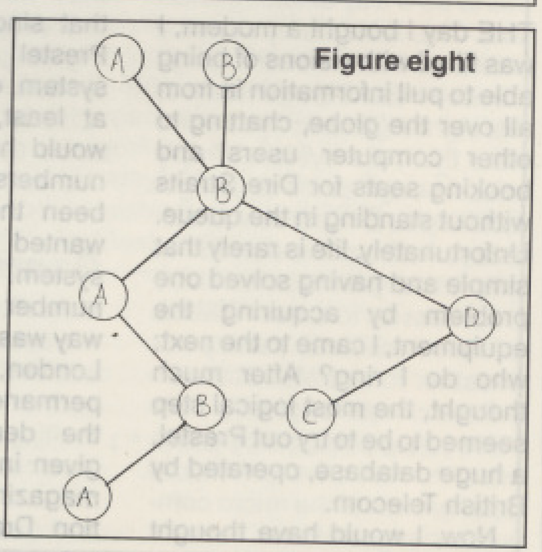
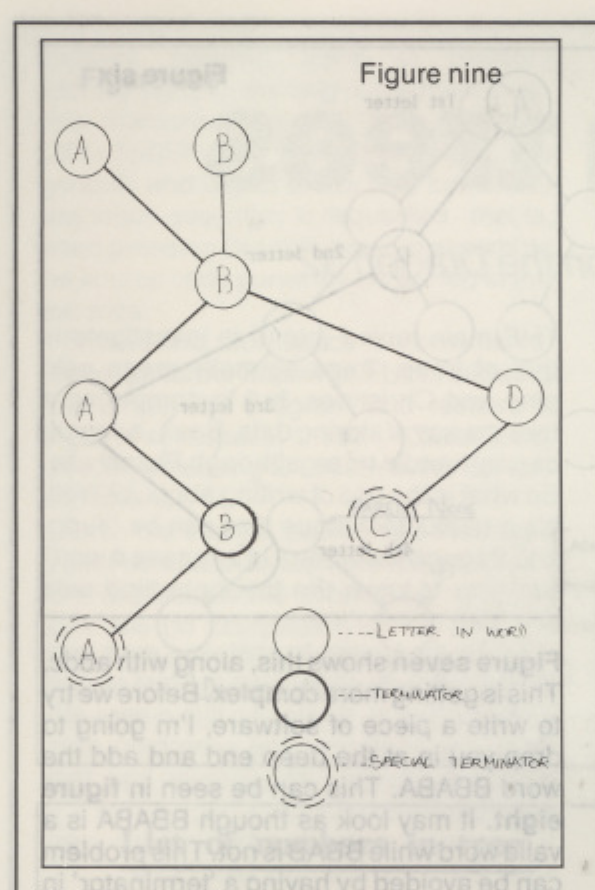
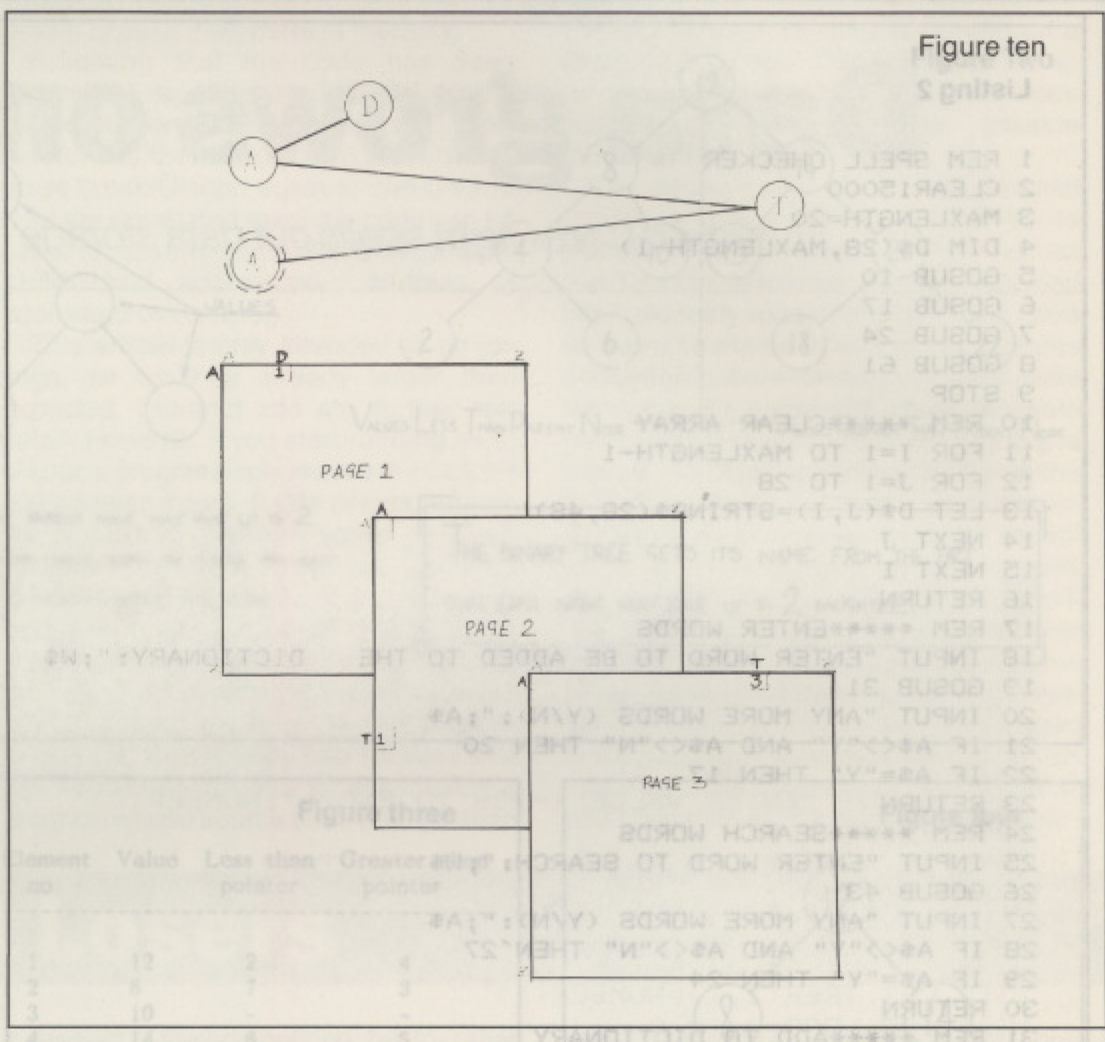


Figure eight





This system allows you to store more words than you would normally be able to do in 32K.

All that remains is for me to reveal how I'm going to 'fudge' this n'ary tree. I can best explain that by a demonstration. Let's take the word DATA. I'm sure you can now imagine how this is going to look as a tree. but for those with a limited imagination I have included it in **figure ten**. Also in this busy drawing are three 'pages'. It is using these 'pages' that I create my tree. On the first page you will see that there is a number 1 at co-ordinates (D,A). D and A are our first two letters, so that explains the location. The 1 means that D followed by A is allowed in a word. The second page has a 1 at (A,T). Again this means that A followed by T is allowed. On the third page we have a three at (T,A).

The three means that this is the end of the word, it is one of our 'special terminators'. So, to recap, in our dictionary we are allowed D followed by A followed by T followed by A to end. This spells the word DATA. Now we could add all the other thousands of words in the dictionary. I fear, however, that our beloved editor might frown upon the idea as she doubtless wants something else in this month's issue. (*At this precise moment, dear people, one issue of Dragon User full of Mr. Cartwright's words would be the answer to my prayer. However, you, the readers, might not be too happy about that and, on reflection, nor would I in the morning. So onwards ...*).

And now for the listings! Yes, that 70 line listing entitled **listing two** is my spell checker! And you all thought it would be a six page job, didn't you? Well, this shows you how efficient data storage can shorten your programs and NOT your lifespan. You can enter words into the 'dictionary' and subsequently search for them.

At the moment, the program allows words of up to twenty letters in length, although this can be altered, memory permitting. If the word that you're looking for isn't there then you are given the option of adding it. You might like to try to get the program to load in your word processed files and let you know which words are not spelt correctly. I've not done that because a) I'm too lazy b) haven't got the time and c) if I wrote the routine for *Electronic Author* then all of you with *Telewriter* will kill me for neglecting you.

The Dragon on Prestel

Ken Smith introduces BT's Prestel database

THE day I bought a modem, I was filled with visions of being able to pull information in from all over the globe, chatting to other computer users and booking seats for Dire Straits without standing in the queue. Unfortunately, life is rarely that simple and having solved one problem by acquiring the equipment, I came to the next; who do I ring? After much thought, the most logical step seemed to be to try out Prestel, a huge database, operated by British Telecom.

Now, I would have thought that since B.T. operate both Prestel and the telephone system, either the operator or, at least, directory enquiries would have a list of access numbers. That might have been the case if B.T. really wanted people to use the system. In reality, the only number I managed to get this way was Prestel head office in London, a number which is permanently engaged. Even the demonstration number, given in one of those inferior magazines which never mention Dragons, proved to be useless. All this time I knew that all the local access numbers were listed on Prestel but would only be available to me once I had made contact. Eventually I found what I was looking for by logging on to a bulletin board and pleading for help.

Love/hate

That was how my love/hate relationship with Britain's biggest database began. A lot of data has flowed along the bus since then. I have now become a regular subscriber to the system and have come to know its pains and its pleasures.

Information is displayed in the form of pages on a viewdata system which allows the use of text and graphics on the same page. Unfortunately, us dragoneers have to be content with a monochrome display since only PMODE4 gives a high enough resolution. To be sure of using a local number you will need a V23 (1200/75) modem, V22 (1200/1200) is being phased in

Continued on page 20

Screen Compressor

***Jonathan Cartwright** squeezes screens into smaller spaces.*

HAVING seen screen compressor programs on much larger computers than the Dragon I decided that it might be a good idea to write one. I'm surprised that it hasn't been done before, as when you've only got 32K to play with, 6K for a hi-res screen is a large chunk of memory to write off.

Before I start with a program, it might be a good idea to explain how a compressor works. Generally speaking, there is quite a lot of blank space on a graphics screen. Obviously, there is less if your picture is complex. This vast amount of data is very wasteful on memory. For example, you could draw on the top half of the graphics screen, and it would still take up 6K. What my program does first is to 'read' the screen. What it is looking for is a selection of bytes, one after another, which are the same. If it doesn't find such data, then it merely stores the screen byte for byte elsewhere in memory. However, if it DOES find what it is looking for them it can get to work. What it does is store a dummy or 'rogue' value to signify that the next few bytes relate to compressed data. The dummy value it uses is the number 255. After this, it places the value of the bytes which are the same. After that, it places the number of bytes which are the same.

The compressed screen is placed elsewhere in memory so that it can be called up, or de-compressed later. I have made the screen be stored at locations 26623 onwards, but you can put it anywhere you like, so long as you CLEAR space and alter the programs accordingly.

The compressor program itself is written in Basic. This is partly for my convenience and it also means that the program is easily readable. To use it, once you've typed it in, LOAD up the screen you wish to compress. Then RUN the program. It will, after quite a long time, tell you the start and end addresses of the compressed screen, along with the percentage saving and the length of the screen in bytes. Because I use a disc drive I have defined the top of the graphics screen as location 3072, and the bottom+1 as 9216.

To use this program without DOS attached, subtract 1535 from these values. Remember to alter the machine code decompressor too.

The next program is a machine code subroutine which does the job of putting the screen back together again. Again it assumes that the compressed screen is at location 26623, but this can be altered to suit your requirements. I have located the routine at 9216, but you can put it anywhere you like. The routine is simply EXECed into action.

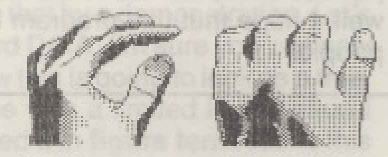
If you want to get adventurous then you could store several screens at different locations in memory, and run a 'slideshow'. This would simply involve changing the store location used in the machine code, before calling up each screen.

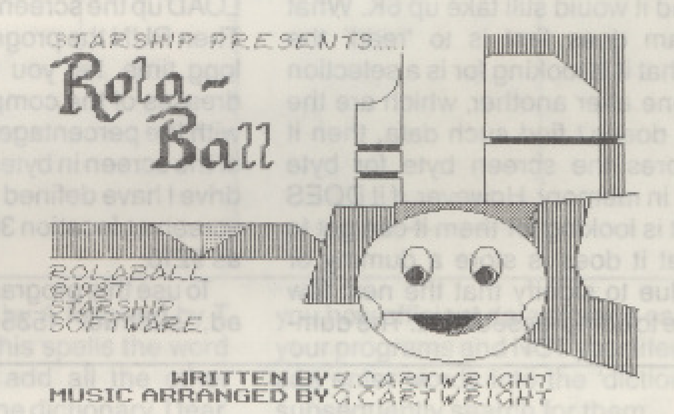
The compressor program can make some fantastic savings, although I have sometimes got savings of as little as 4 per cent. My letter-head now takes up nearly half the space it used to, and if I save compressed screens to disc, instead of expanded ones, I can save memory there as well. I hope that this program is of use to people.

```
4 CLEAR200,26623
5 REM **SCREEN COMPRESSOR
6 REM **(C)1988 STARSHIP SOFTWARE
10 PMODE4,1:SCREEN1,1
20 LO=3072:REM TOP OF SCREEN
21 L1=9216:REM BOTTOM OF SCREEN
30 ST=26623:REM STORE LOCATION
35 S1=ST
40 IF PEEK(LO)=PEEK(LO+1) AND PEEK(LO)=PEEK(LO+2) THEN GOSUB 1000:GOTO100
50 IF PEEK(LO)=255 THEN GOSUB 1000
60 POKEST,PEEK(LO)
70 ST=ST+1
80 LO=LO+1
100 IF LO<L1 THEN GOTO 40
110 GOTO 2000
1000 NB=0:REM NUMBER OF BYTES THAT ARE THE SAME
1010 L2=LO
1020 IF PEEK(L2)=PEEK(LO) AND NB<255 THEN NB=NB+1:L2=L2+1:GOTO 1020
1030 POKEST,255:REM DE-COMPRESS VALUE
1040 ST=ST+1
1050 POKEST,PEEK(LO):REM VALUE OF BYTES
1060 ST=ST+1
1070 POKEST,NB:REM NUMBER OF BYTES
1080 LO=LO+NB:ST=ST+1
1090 RETURN
2000 PRINT "SCREEN COMPRESSED"
2010 PRINT "START ADDRESS:";S1
2020 PRINT "END   ADDRESS:";ST-1
2030 BU=ST-1-S1:REM LENGTH OF SCREEN IN BYTES
2040 PRINT "BYTES USED:";BU
2050 PRINT "SPACE SAVING:";100-(BU/6144*100);"%"
2060 END
```

```
(C)COPYRIGHT COMPUSENSE LTD 1983 DASM 2.00
   30      ALL
6B01              30      FML
2400              40      ORG 9216
2400 BE0C00       50      LDX £3072
2403 108E67FF     60      LDY £26623
2407 A6A0         70 @GETB LDA ,Y+
2409 81FF         80      CMPA £255
240B 260D         90      BNE @NORMAL
240D A6A0        100      LDA ,Y+
240F E6A0        110      LDB ,Y+
2411 A780
2413 5A
2414 C100
£2416 26F9
2418 2002
241A A780
241C BC2400
241F 25E6
2421 39
2422
   120 @DISP STA ,X+
   130      DECB
   140      CMPB £0
   150      BNE @DISP
   160      BRA @NEXT
   170 @NORMAL STA ,X+
   180 @NEXT CMPX £9216
   190      BLO @GETB
   200      RTS
  1000      END
```

Starship
Software
23 Tintern Road,Cheadle Hulme,Cheshire,
SK8 7QF.





The above letter-head has a space saving of 47.4%

My Rola-Ball screen boasts a respectable 28.2% saving.

Continued from page 18
and is already available on many numbers. Where V22 is available the system will automatically adjust to match your equipment so the only way to find out is to try. The system is so large that it is easy to get lost, so much so that B.T. find it necessary to provide a directory, listing all the information providers (IPs) as they are called. When a system this big is largely menu driven then it can take a long time to get to where you want to go, so there has to be a way of taking short cuts. This is normally done by entering the relevant page number. Now a page number has eight digits and if you enter less, then the system simply adds enough zeros to bring it up to eight. Of course you can always look up the number in the directory, but wouldn't it be much easier to enter *EMAIL for the mailbox facility or *weather for the Met Office.

Prestel was originally conceived as a business database. However, the advent and rapid growth of the micro computer changed all that. A whole new market appeared and it was decided to expand the network to exploit this. There are now bulletin boards to cater for a wide variety of interests, many of them having nothing to do with computers. For instance, one section deals with pop music and allows you to have your say. If you think Rick Astley's records are a pain in the earholes then you can say so and leave your views for all to read. Deaf users (*ex-Astley listeners? — Ed.*) have their own section, known as 'Earmail', where they can publicise matters pertaining to the hearing impaired. I have even found a religious section run by 'Christel' and yes, you can book tickets for concerts at Wembley.

Game changes

Most of the games available on the system are charged for in addition to the subscription and time charges, Shades (a multi user dungeon game), for instance, costs an extra 2p per minute. Many of the games and quizzes offer cash prizes but it is my experience that they are more designed to take your money than to entertain.

Working as I do, at the sharp end of a service organisation, the thing that I find most useful is the telex link. Most companies have a telex machine and using this system, my company can leave me a day's work and I can call it off at my convenience. (Where do you keep your Dragon?) I can send my reply that evening and it will be waiting for them in the morning. Telex link is part of Mailbox, the electronic mail system. This was apparently included as an after-thought. Perhaps this is why it is a bit crude, for instance no word-wrap, no real edit facility and typing at a transfer rate of 75 bits/second can be painfully slow.

Subscribing to Prestel will set you back £6.50 a quarter. Think of it as fifty pence a week and you will see it is not bad value for money. During peak usage times, ie 9am to 6pm, there are additional charges of 6p per minute; outside these times there are no time charges. There are additional charges for some services but there is adequate warning before you incur any additional cost. To get a real taste there are free demonstrations available on the numbers listed below. When prompted for your ID number, enter 4444444444, then for your password use 4444. This should get you started, so why not give it a try. It will only cost you a phone call.

Finally, listed below are numbers that are supposed to give the demonstration. At the time of writing the only one that does not work is the London number. If you live in the south and cannot get any joy, then try 0227 455400. Best of luck and if you do subscribe, why not give me an MBX on 819992400 and tell me what you think.

Prestel access numbers

South (01) 618 1111
Midlands (021) 618 1111
North (061) 618 1111
Scotland (041) 618 1111

Copy to order

***Rudy Duyck** presents a versatile file copying program for Dragon Data and Eurohard DOSses.*

COPY is a utility to copy any number of files any number of times from one drive to another, and works with all Dragon Data and Eurohard DOSses, including Super-DOS. It is written in Basic with one very short machine code routine (relocatable) used to scroll the screen downwards and stored in the string MC$ in line 15, using the data in line 1500. Because of this it is impossible to use the program as it stands with any form of high resolution text screen. COPY makes its own directory of the source disc and offers you the options of copying all files, a number of selected files, or all files except for a number of selected files. Selection of files is straightforward through the use of a cursor positioned with the up and down arrow keys. The order of copying files can be determined in this way too. I found the information needed to write this program in Grosvenor Software's *DragonDOS: a Programmer's Guide*. If you want to understand how COPY creates its own directory, you will also find excellent information in Paul Dagleish's article *Into The Directory* in the May 1987 issue of *Dragon User*.

The COPY utility will ask you to enter the number (1 or 2) of both the source and the target discs (drives). These must be different: it is unfortunately impossible to copy files from one disc to another using only one drive under DragonDOS. After being given this information COPY will access the source disc and create its directory in the array NM$. Only valid files will be included.

When the end of the directory is reached (found by checking for the end of directory flag in line 150,) the Dragon produces three BEEPs to warn you and gives you three options: copy all files, select the files you want copied, or select the files you want omitted. Next you are aksed whether you want all the files copied to be protected. The default option, also selected if you press ENTER and indicated with a star, is no protection. If you selected the second copy option you will now be able to determine the order in which files are to be copied. With both the second and the third options you get the first screen of the directory and a cursor which you can direct with the up and down arrow keys (which autorepeat). You select the files by pressing the space bar and you end the selection process and begin the copying with ENTER. Pressing "#" will erase all the selections you have made so far and enable you to start all over again if you have made a mistake. The screen will scroll up or scroll down as needed.

After the copying is done you are asked whether you want to copy the same files once more, to copy different files from the same disc, to copy files from a different disc, or to stop. If you want to copy from the same disc the program saves time by using the directory already in memory. All the copying takes place in one go and COPY will warn you with three BEEPs when it is finished so that you can do something else during the copying (it does not take very long, though). Should any error (except a DISC or DIRECTORY FULL error) occur, the copying ends and you get an error message. If the error occurs because the target disc is full (DF error, code 148) or because its directory is full (FD error, code 146) you will be asked to insert a new disc for the other files to be copied onto. The most likely only partially saved file last copied is killed to clean up the full target disc.

You can use COPY as a faster alternative to BACKUP (especially in all versions of DragonDOS after 1.0: they are slooow!) by using a blank formatted disc as target disc and selecting the copy all option. This makes it also possible to backup between different disc formats. If like me you have the standard Dragon Data disc drive unit with a higher-capacity drive added (in my case a double-sided, 40/80-track switchable drive) this comes in quite handy! It is also possible to have all files protected in one go.

COPY enables you to re-organise discs by copying files from one cluttered up disc to a newly formatted disc in a different order. Even if you choose to copy the files in the same order as on the original disc you should be able to save space because often-used discs are not especially frugal in their use of space. You may wonder why I have not built in an option to sort file names alphabetically. The reason is very simple. I do not see the use of this since quite often a particular software package consists of several programs or includes data files and all of these would end up scattered all over the disc and the directory if you sorted the files alphabetically. In this way the logical connection between the various components of a package would be lost.

It is easy to merge existing discs onto a blank one using COPY too. A final use for this utility is for people, eg members of a computer club, who want to distribute programs they have written to a number of fellow members, or who want to circulate their newsletters on disc (as some clubs do). You can do this far more easily and in a less time-consuming manner with COPY because the program allows you to copy the same items time and again.

For those Dragon users who are interested in the way COPY is programmed let me draw attention to a number of features which may provide inspiration for your own programs. The routine in lines 50 to 230 to produce a directory can quite easily be adopted and adapted in your own programs that are to manipulate disc files, eg to run, rename, or delete programs *en masse*. The routine in lines 830 to 870 checks for keys being held down without using INKEY$. The major advantage of this use of the keyboard roll over table in lower memory (336-345) is that the keys autorepeat. A final remark concerns the error routine in line 1000. Most people seem to believe that error trapping in only useful for preventing wrong user input from interrupting a program. However, error trapping can also be used to advantage in disc operations, eg to prevent a wrong filename or full disc from aborting the whole program. If you divert the program from an error to a routine that can diagnose the error and take action accordingly you produce a program that is often more user friendly and that can deal with disc errors that occur frequently when handling files. In COPY the error routine handles the disc and directory full errors that one must expect to occur when copying from a larger-format disc to a smaller-format disc for instance.

I shall answer any questions in connection with COPY you might have if you include a self-addressed envelope and one International Reply Coupon if you live in the EEC or two if you live outside it. If you do not feel up to the job of typing in the listing, I can send you a disc with the program on it, if you send me 270 Belgian francs by International Money Order (no cheques or postal orders please: the costs deducted by the bank are too high for a small sum paid this way). My address is: Rudy Duyck, Abdijbekestraat 8-C2, B-8200 Brugge 2 (St. Andries), Belgium.

```
5 'copies files from drive br to drive dl
6 'copyright rudy duyck 15.03.87
10 PCLEAR1:CLEAR10000:ERROR GOTO1000:DIM NM$(160),VL(160)
15 FOR I=1 TO 17:READ N:MC$=MC$+CHR$(N):NEXT I:GOTO30
20 H1$=RIGHT$(STR$(BR),1)+":"+NM$(I):H2$=RIGHT$(STR$(DL),1)+":"+NM$(I):GOSUB25:P
RINT
21 COPY H1$ TO H2$:IF PR THEN PROTECTON H2$
```

continued

```
22 CLOSE:RETURN
23 PRINTUSING"####";VL(I);:PRINT" ";
25 PRINTUSING"%          %";LEFT$(NM$(I),LEN(NM$(I))-4);:PRINTRIGHT$(NM$(I),4);:RETU
RN
30 CLS:INPUT"SOURCE DISC";BR:IF BR<>1 AND BR<>2 THEN 30
40 PRINT:INPUT"TARGET DISC";DL:IF DL<>1 AND DL<>2 THEN 40
50 N=1
100 FOR SR=3 TO 18
110 SREAD BR,20,SR,H1$,H2$
120 H$=H1$+LEFT$(H2$,127)
130 FOR I=0 TO 9
140 VL(N)=ASC(MID$(H$,I*25+1,1))
150 IF VL(N) AND 8 THEN I=9:SR=18:GOTO220
160 IF VL(N)<>0 AND VL(N)<>2 AND VL(N)<>32 AND VL(N)<>34 THEN 220
170 NM$(N)=MID$(H$,I*25+2,8):K=INSTR(NM$(N),CHR$(0))
190 IF K THEN NM$(N)=LEFT$(NM$(N),K-1)
200 NM$(N)=NM$(N)+"."+MID$(H$,10+I*25,3)
210 N=N+1
220 NEXT I,SR
230 N=N-1
250 FOR I=0 TO N:VL(I)=0:NEXT I
255 BEEP3
260 CLS:PRINT@64,"0. COPY ALL FILES":PRINT"1. SELECT":PRINT"2. OMIT":PRINT:PRINT
:INPUT"OPTION";VL(0)
270 IF VL(0)<0 OR VL(0)>2 THEN 260
280 CLS:INPUT"PROTECT FILES (Y/N*) ";A$:IF A$="Y" THEN PR=1 ELSE PR=0
290 ON VL(0)+1 GOSUB 500,600,700
300 BEEP3:CLS:PRINT@64,"1. COPY SAME FILES":PRINT"2. SAME DISC":PRINT"3. DIFFERE
NT DISC":PRINT"4. STOP":PRINT:PRINT:INPUT"OPTION";I
305 IF I<1 OR I>4 THEN 300
310 IF I=1 THEN ID=1 ELSE ID=0
320 ON I GOTO 290,250,30
330 END
500 CLS:I=1
510 FOR I=I TO N:GOSUB 20:NEXT I:RETURN
600 IF ID=0 THEN GOSUB800:IF NN=0 THEN RETURN
610 CLS:T=1
620 FOR I=1 TO N
630 IF VL(I)=T THEN GOSUB20:I=N
640 NEXT I
650 IF T<NN THEN T=T+1:GOTO620
660 RETURN
700 IF ID=0 THEN GOSUB800:IF NN=N THEN RETURN
710 CLS:I=1
720 FOR I=I TO N
730 IF VL(I)=0 THEN GOSUB20
740 NEXT I:RETURN
800 CLS:PRINT@160,"ENTER    = DONE":PRINT"#        = REDO":PRINT"SPACE    = SELECT/
OMIT":PRINT"UP/DOWN = MOVE CURSOR":PRINT:PRINT:INPUT"PRESS ENTER";A$
810 CLS:IF N<16 THEN NN=N ELSE NN=16
815 FOR I=1 TO NN-1:GOSUB23:PRINT:NEXT I:I=NN:GOSUB23:I=1:NN=0:CU=0
820 PRINT@CU,">";
830 IF PEEK(342)=223 AND I<N THEN PRINT@CU," ";:I=I+1:IF CU<480 THEN CU=CU+32 EL
SE PRINT@511:PRINT@CU,"";:GOSUB23
840 IF PEEK(341)=223 AND I>1 THEN PRINT@CU," ";:I=I-1:IF CU>0 THEN CU=CU-32 ELSE
 DN=VARPTR(MC$):DN=256*PEEK(DN+2)+PEEK(DN+3):EXEC DN:PRINT@CU,"";:GOSUB23
850 IF PEEK(345)=223 AND VL(I)=0 THEN IF VL(0)=1 THEN NN=NN+1:VL(I)=NN:PRINT@CU,
"";:GOSUB23 ELSE VL(I)=-1:NN=NN+1:PRINT@CU,"";:GOSUB23
860 IF PEEK(337)=190 AND PEEK(341)=254 THEN FOR I=1 TO N:VL(I)=0:NEXT I:GOTO800
870 IF PEEK(338)=191 THEN RETURN
880 GOTO 820
1000 ERROR GOTO1000:IF ERR=148 OR ERR=146 THEN PRINT:PRINT"USE ANOTHER DISC FOR
REMAINING  FILES":KILL H2$:CLOSE:PRINT"NEW TARGET DISC IN DRIVE";DL;:INPUTA$:ON
VL(0)+1 GOSUB510,620,720:GOTO300 ELSE PRINT"ERROR";ERR;" IN LINE";ERL:CLOSE:END
1500 DATA &H8E,&H05,&HE0,&H10,&H8E,&H06,&H00,&HEC,&H83,&HED,&HA3,&H8C,&H04,&H00,
&H26,&HF7,&H39
```

Write: ADVENTURE

Pete Gerrard generates an error message

THERE can be no greater crime for the adventure writer than to release a game that hasn't been fully tested onto the unsuspecting public. People who have paid good money for a program have a right to expect that program to be working correctly. A bad pianist will not make a living, a plumber who doesn't do his plumbing properly will not get paid, and equally so the adventure writer who doesn't care to check a game does not deserve the support of the adventure playing public.

And yet, games released by the largest of companies often contain an unenviable collection of mistakes. It is not the fault of the company handling the distribution, since they will be paying money in one fashion or another to the company or individual who wrote the game in the first place. Under contract, more often than not, and contracts are meant to be legally binding. Thus if the programmer leaves errors unchecked then not only is he treating the public with contempt, if the contract contains some such phraseology as "will deliver a fully de-bugged, error-free adventure" then he is also breaking the law.

Thus it is of vital importance to have your games checked and playtested before either selling them yourself or submitting them to the company who asked for them. This should preferably be done by someone who knows nothing about the game, and while the initial and final checking should be done by the programmer the all-important middle stages should be handled by someone else. Like proofreading an article such as this, for example. I know what it contains, and am therefore likely to miss the odd mistake or two. In other words, I will probably see the word I want to see rather than what is actually written down. Likewise, the programmer will almost inevitably play the game as it is meant to be played, and not go down all those wonderfully obscure paths that make the playing of a competent adventure so enjoyable.

Seized up

With the ever-increasing complexity of adventure games those obscure paths can lead to mistakes that are often hilarious if caught before the game is released, but a death trap if they are seized upon by a reviewer in one of the popular magazines. Such a reviewer could well be my brother Mike or I, and on one of the games that we produced by working as a team we made some marvellous mistakes. All, fortunately, caught before the game was actually released.

It was a two part adventure based on Homer's epic *The Odyssey*. Mike did the design for the game, and I did the programming, and we tested it between us. This worked quite well, for he didn't know how the program was put together and I was just following instructions blindly and thus didn't know the correct route to get from problem (a) to problem (b). In order to fit everything into memory I was doing some text compression. This involved tokenising words and commonly occurring groups of letters, so that one character replaced three or more. Thus 'from' would be represented by one token, 'ard' by another, and so on. Mike was rather astonished in one early version of the game while he was exploring a king's palace and went somewhere where he shouldn't have. Immediately he was 'surrounded by several large gufroms'. Ah, the penalties of getting two tokens mixed up. Wonderful word though gufrom might be, a sort of small, hairy creature, I would imagine, it would not do for a reviewer to find a gufrom marauding through his copy of the game.

Flags, counters, call them what you will, they play a vital part in any adventure. A flag incorrectly set, or, just as bad, not set at all, can produce a minor disaster. In order to escape from the Cyclops' cave in The Odyssey our intrepid hero has first of all to blind his captor and then climb under the belly of a sheep and get out as part of a herd of the things. The blinded Cyclops only feels the tops of the sheep and fails to find Odysseus clinging grimly on underneath one of them. When he has successfully escaped he can then tell the sheep to go away, and carry on the game. Alas and alack I forgot to set the 'sheep gone away' flag, and so wherever Odysseus went he was followed by 'a flock of sheep are grazing close by you'. He could scale a castle wall, and there would be a flock of sheep by his side. He could swim to his ship, and find a flock of aquatic sheep grazing on the water. That was soon corrected, but not before a subtle hint from Mike had put me onto the problem. It was something like 'I can't get rid of these ****** sheep!' Literary talent will out, I always say.

A brick dropped

The role of a playtester is not an easy one, and I always think that they should be highly rewarded if they perform a competent job and successfully eliminate all errors. One of the most common mistakes by the programmers seems to be the object that can never be dropped ...

⟨DROP BRICK
Okay, you drop the brick.

⟨INVENTORY
You are carrying a brick.

⟨DROP BRICK
Okay, you drop the brick.

And so on, for ever and ever. A little flag not getting set somewhere. Likewise we have objects that can be continually taken, and yet are always present in the room description. Conversely, we have objects that are there, and yet you are told they are not there when you try to take them. Again, objects that vanish into limbo when you drop them and which are never seen for the rest of the game. The playtester must attempt everything possible with every object in every location, in an ideal world, and if the game in question doesn't contain a save feature then send it back to the programmer with the politely worded request that you will not test the game until it does.

SAVE and LOAD are two commonly used words that easily lead to errors, and thus I prefer to have SAVE and RESTORE, making sure that the parser recognises much more than the first four letters of each word if possible, or have a special condition for the word RESTORE on its own. Players might want to have a rest, not restore a previously saved game. They might want to load a revolver, not load an old file. Once more, the playtester must weed these out.

I was fortunate enough on one game to have a really efficient playtester. The game had originally been written on one computer by my brother Mike and I, and then I transferred it over to another one. This meant that it needed testing all over again, and so out it went, with the inevitable sinking feeling that I might, after all, have overlooked something. Needless to say, I had. The game concerned those two redoubtable characters Holmes and Watson (now out of copyright and open to anyone to write an adventure about), and featured a dog called Toby. On examining Toby you were told that you could see some fleas. Examining the fleas told you that you could see some smaller fleas. Having a bit

more memory left on the second computer than the first one I continued this examination of fleas to a never-ending degree, so that examining the smaller fleas led you to be told that you could see some that were even smaller. This went on and on, and thank goodness for the competent playtester. He examined more fleas than I had ever imagined possible, with the end result being that the machine finally ran out of memory and crashed. That was definitely a case of having too many bugs in a game!

Never be tempted to put anything like that into your games, no matter how much memory you might have left, without putting some kind of limit on things beyond which the fleas will not get any smaller, or whatever. Someone, somewhere, will will feel flea and find an error.

When it comes to searching out errors, which after all is what a playtester should be doing, one of the things that is often overlooked is the very first move of a game. Adventure players are a contrary lot, and on that first move have you checked for someone typing in OOPS? Or AGAIN, if you've allowed them to repeat a command? Or RESTORE from RAM when there isn't a previously saved position? A program crash on the first input will not endear anyone to your adventures.

Removing every mistake is, I would imagine, impossible in a highly complex game. You are just not going to be able to foresee all the possibilities, and that is why you ignore playtesting at your peril. A good reviewer will spend many hours on an adventure before writing his or her review, and if they find anything within that time that strikes them as an error, particularly if



it's a crucial one, then believe me that is what the reviewer will concentrate on. I know, I've done it myself! It's always difficult to come up with something new when writing a review, but a critical mistake is a godsend. Don't give anyone the opportunity to lambast your game by leaving in silly errors.

Something that is often overlooked, and again here a playtester is essential, is your spelling and grammar. The incorrect use of ITS and IT'S, THERE or THEIR, is irritating. Always have a dictionary by your side when writing or testing a game, and any doubtful words should be consulted straight away. It's (ha ha!) always difficult writing about grammar, because someone will pick you up on the inevitable errors, but adventure games should content themselves with sticking to the simple things and getting them right. Try and strike a balance between 'You are on a road.' and a Bernard Levin-type 500 word room description that contains just one sentence.

The simple things are always the best. 'You are crunching noisily along a gravel path' is far better than 'You are making a lot of noise as you walk along a path with a lot of gravel on it. That's what is making the noise, the gravel, as you walk along it, on the path.' I've seen them, I've seen them, honestly! A good playtester will tell you off for writing something like the latter example, and quite right too.

A good playtester is invaluable, and you should never, ever, sell or submit an adventure until it has been tested by at least one person other than yourself. You know how a game should be played, but the great public that control your financial return do not, and they have a right to an error-free game. Test it!



I was going to devote this month's column to one of my 'story' solutions, involving our old friend Professor Deadrock exploring a little something called the *Pyramids of Doom*: an adventure that many people seem stuck on (or is that in?).

Steam up

However, a letter arrived in tandem with an adventure program written by the author of the letter, who wanted the adventure reviewed within these sacred pages. Perhaps it will be reviewed by someone else, I don't know, but in these days of inactivity by Dragon software houses there are certain ways to go about giving a game a reasonable chance of getting reviewed. There are also certain ways of ensuring that an adventure will not be reviewed. The author of this letter has taken the latter course, and I hope that all you adventure fanatics out there will bear with me for a while as I digress into the worlds of right and wrong.

If I am asked to review a game of the adventure variety I like to see at least a partial map and solution. The time involved in exploring a new game is such that, unless the game proves to be particularly wonderful, it's not worth doing. Reviewing an adventure game takes a good few hours (or days!), and without a helping hand you are in the same position as someone who has just bought it off the shelf or ordered it through the post. Having said all that, I will still look at a game as long as the accompanying letter is at least polite.

A good number of years ago I converted three adventures that I'd written so that they would work on the Dragon. I had little knowledge of the computer, but my publishers insisted on the conversion as the adventures were accompanying a book. Three listings were given in that book, but the publishers wisely wanted a tape as well, in case people couldn't be bothered typing in page after page of listings. Fair enough, and I spent a reasonable amount of time getting these three adventures up and running on the Dragon. To send a game in for review that has been recorded over one of my old tapes is not the best way of endearing yourself to the person who is hopefully going to review the game.

Exclaim disclaim

The letter that came with this aforementioned tape gave me a brief summary of the plot. Nothing exciting there, but perhaps a great adventure lurked beneath the bald

outline. After this summary came the quote 'a brief and simple adventure by my standards!' Okay, perhaps the chap is a mite big-headed, but then we all could fall into that particular category at times. I am told that someone else is 'hooked on her early review copy!' So? I haven't seen the game yet.

The exclamation marks are all in the letter, by the way, they are not mine.

I am told that the adventure features a VLIST command that gives a list of verbs used by the game. Good idea. Such verbs as 'verbose & brief' are quoted, along with the author's comments 'rather nice additions!' Infocom have been doing it for years, albeit not on a Dragon.

Another quote: 'Danger situations exist, prompted by 'What should I do?', and only one response works!' If there is anything that I hate in an adventure game more than anything else it is the instant death situation. It is so pointless. Adventures are meant to be a test of a person's logic and, perhaps, lateral thinking, and to find yourself being killed off for no apparent reason other than that the parser cannot understand a particular sentence structure irritates me in the extreme. One response in an instant death situation? Come on, there are better puzzles to be made than that.

I shall quote again: 'Two maze areas exist, the mine area and the marsh. The mine needs mapping as it has loads of blind areas with no exits and all the exits are hidden; there being three I think!' I dislike mazes at the best of times, but can see their point in a well-structured adventure and have indeed spent many an enjoyable hour working my way through a particularly difficult maze. However, blind areas, no exits, exits hidden? An easy way of quickly programming a difficult adventure.

Get it write

Finally I get the comment 'I hope you'll give it a fair bit of space in D. User, please, as I have not been mentioned for months ... (obviously not been reading the reviews in *Dragon User* — P.G.) ... XXX Software need a boost as they cannot afford D. User advert rates!' You have not been mentioned for months, and you will not be mentioned unless I get a reasonable letter. Give it a fair bit of space? There's nothing better than free advertising, you know who you are, so climb off that pedestal and admit that other people are of interest in the adventure world as well as yourself.

If you (or for that matter anyone else) would like a game reviewed here, then please send me at least a partial map and solution, don't demand acres of column space, don't put pointless puzzles in it, and above all don't boast.

And now that I've got all that off my chest, I shall climb down from my pedestal as well and, somewhat red in the face, continue with the Adventure Trail

Space now forbids a story solution to *Pyramids of Doom*, so as this is being written in a rush (editor moves deadline forward two weeks. Either that or failed to get last month's copy ... leaves space for comment from She Who Must Be Obeyed ...(*OK, OK, I apologised! Next time be a dear and put it in an envelope more than two inches square? Please?*)So we shall have nothing less than the full solution, courtesy of the admirable Martin Edwards and Richard Stedham:

1) Get the pole (a shovel) and GO POOL. Get key, get liquid (in canteen) and E, N, E; DIG, get the small key, S, DIG again and GO HOLE. Unlock the door then return U again. Drop the shovel, get the stone. Unlock the main door and go in, then drop both keys and light the flashlight. You are in the Pyramid of Doom.



2) Open the sarcophagus. Go N, E and get the flute. Go W, S, S and play the flute here. Examine (use LOOK) the fireplace to find a coal (thinks: did Magnetic Scrolls play this idea through and use it in Guild of Thieves?). LOOK fireplace (some ashes) and examine ashes to find a necklace. Get the NECKLACE. Go passage then go N and E. Read the hieroglyphics (if you can spell them!) and drop the stone and the necklace. Get the jerky (what a wonderful thing).

3) W then N and drop the jerky. The oyster will eat the jerky, dropping a pearl, so you should get the PEARL, go archway, E, drop pearl, W, N, and also you should drop the flute here, too.

4) Go W, S and go sarcophagus. D, clean the coal (it's a ruby, but isn't a treasure really) and pour water to extinguish burning leaves. Now you can get the TAPESTRY. Go alcove, look box, look box (yes, again!) and get the iron glove. Get the skull and examine it. Get its GOLD TEETH.

5) W, S, and drop the skull. WOW! N, N, and wear the glove. You'll probably drop it (sweaty hands, you see) so pick it up and re-wear it. Hit doorway and remove the glove.

6) Get the rope and go S, S, go ladder, then S again. Throw rope (up to the ceiling to help you get up there) and throw ruby (to help kill the pharoah) then get the SAPPHIRE.

7) S and LOOK RUBBISH then look at the explorer too. Get the PIN and the CARVING then go W and D. Drop the canteen and get the saw.

8) N, U, W, N. Saw table, drop saw and get NECKLACE. E and go archway. Drop the tapestry, teeth, necklace, pin, carving and sapphire. Go W and N.

9) W, get saw, S, get key, get key. Go sarcophagus. D, S, go ladder, S, go rope and open the chest. Get the CROWN. Pull the chain. Go stairs, saw bars, drop saw, wear glove, unlock coffer, drop both keys. Look coffer and get the BRACELET. Go window, get BAR, go W, D, D, W, D, N, N, go door, feel floor and get COIN. Go E, light flashlight and get the SCARAB. Go W, W, light the flashlight, S, U, W, N, E, go arch, E, drop scarab, coin, bar, crown and bracelet.

10) SCORE.

Adventure mags

And there you have it, chaps and chappesses, the complete solution to an adventure that conjures up some rather obscure problems at times, but all should now be revealed in order to enable you to file away another game as 'completed'.

Finally for this month we have two changes of address to note (apart from the obvious one). I've mentioned before in this column a couple of home-produced adventure magazines known as *Soothsayer* and *Adventure Probe*. Both were set up by the hard-working Sandra Sharkey, who deserves much praise for all the effort she put into launching them and getting them onto a regular monthly basis. If you want complete maps and solutions, *Soothsayer* is the one to go for. On the other hand, if you prefer the chattier approach with some hints, tips, reviews, stories, a veritable potpourri of adventure information, then *Adventure Probe* is the magazine for you. Once upon a time they were both produced in the great metropolis of Wigan (Challenge Cup Winners, well done lads!), but recently, owing to a variety of circumstances, they have moved elsewhere. For *Soothsayer*, you will need to contact John Barnsley at 32 Merrivale Road, Rising Brook, Stafford, Staffs. ST17 9EB. I love the first part of that address, sounds like something out of *Lord of the Rings*, written by that chap whose name I used not to be able to spell. For *Adventure Probe*, your editor is now Mandy Rodrigues (Welsh, despite the name), who resides at 24 Maes Y Cwm, Llandudno, Gwynedd LL30 1JE. A simple enough address when you get used to it! Sample copies are currently available at £1.25 for *Soothsayer* and £1.00 for *Adventure Probe*, if you fancy sampling their wares.

End of space, as usual. To the un-named person who sent the game in for review, don't despair. To that person, and the rest of you, bye for now.

A lifetime in 48 days

How long to make your first million? **says Gordon Lee**

Prize

Reviewed in last month's *Dragon User*, we now have ten copies of **Super Nova** from **Orange Software** for the stars of the July competition.

Rules

It ain't what you say, it's how you say it. Say it with your Dragon: print out your working, add any comments you wish to add, put them in an envelope marked JULY COMPETITION and send the lot to the usual address which is now, let me remind you, Dragon Publications, 49 Alexandra Road, Hounslow, Middx TW3 4HP.

And now the tiebreaker — fie on you who thought I wouldn't have time to think of a tiebreaker. Let's think. Everybody has a favourite number. Or an un-favourite number. Tell us about your favourite (or un-favourite) number. The wittiest one gets quoted.

April winners

TRICKY one, this. However, a fair number of readers had the right answer up their sleeves. The lucky winners are:

E A Newman of Addlestone, S A Siddiqui of Chiswick, Graham Barber of Sutton Coldfield (the man who has everything except a new printer ribbon), R Raine of Sapcote, Don Robertson of Epsom, Patricia Hill of Carshalton Beeches, D Hartley of Cleckheaton, Dave Lardner of Glasgow, Austan Henderson of Bromsgrove and Fred Willers of Yardfield. Denis M. is spreading horrible rumours about the tiebreakers again, but loses by a nose (now, there is a man with a new ribbon).

Some of the tiebreakers were too close to surreal life for comfort. I observed that the best ones were mostly from non-winners. Can Denis be right after all? But no! We are saved. The ribbonless but victorious Barber presents:

'The Editor said: "They'd like their drink better if it were decaffinated". I thought she said "They're likely to think better if they were decapitated." So I said, "Off with their heads".'

'An understandable mistake' said Alice.

Commendations to everyone who mentioned heads, and everyone who mentioned deadlines owes the Editor an aspirin.

The prize this month is a gift bag of their recent releases **Underbeings of Croth** and **Pyradventure** and discount vouchers from **Dragonfire Services**. Selections will be on their way soon.

Solution

See opposite page.

WHEN the British mathematician G H Hardy called on the Indian mathematical genius Srinivara Ramanujan, he is reported to have remarked that the number of his taxi, 1729, was not a very interesting one. "Not so", replied Ramanujan, "It is the smallest number expressible as the sum of two cubes in two different ways!"

While not suggesting that this is typical of encounters between mathmaticians, it does illustrate the fact that some numbers are more interesting than others. But what exactly constitutes an 'interesting' number? Can any number be said to be 'uninteresting'? Of course, a number need not be interesting purely in a mathematical sense. No doubt we all have a personal 'top ten' of favourite interesting numbers — even if they are only the ones we use to do the football pools.

Amongst the lower numbers it is not difficult to find an association with any given number. For example, the number 6 to the biochemist might denote the sixth element, carbon, the basic element on which life depends. To the historian, that same number might conjure up thoughts of the six wives of Henry VIII, or even the six King Georges, while the musicologist would think of 'Les Six', a group of French composers with their own distinctive style. The mathematician would undoubtedly note that 6 is the smallest 'perfect' number — that is, its whole number divisors (1, 2, and 3) also add up to its own value.

But what about 'uninteresting' numbers? Are there any of those? Let's suppose that we make a list of all numbers, starting at 1 and progressing upwards. On the left-hand side of the page we put all of the 'interesting' numbers. This, in its early stages, would be fairly extensive. Sooner or later we would come to a number which, despite all our attempts to find some interesting aspect about it, we would be unable to do. Suppose that this number is 32,769. (We have already placed 32,768 on the left-hand list as being interesting as it is the precise number of bytes that is to be found in a 32K ram such as the one on the Dragon.) However, by placing this number on the right-hand list it now becomes 'interesting' as it is the first number to appear on this side. This compels us to declare that it is, after all, interesting, so we have to delete it from the right and place it on the left side of the page. This leaves the right-hand list vacant once again — at least until the next number is placed there, when it too will need to be transferred. Thus, by this logical (or illogical) process *all* numbers become interesting — no matter how high you care to go.

Mention of high numbers reminds me of the story, supposedly true, of the American teacher who told his class that it was impossible to count up to one million in a normal life span. Whereupon the mother of one of his pupils proved him wrong by not only doing it but also producing the result typed out in full! This, presumably, was in number form rather than in words, but even so (allowing for a space between each number) there would be nearly seven million typed characters in the final list, occupying 1,000 double-sided sheets of paper. Even at a typing speed of five characters a second, working eight hours a day, and allowing no time for thought, the task would take about 48 days. If you have a couple of reams of printer paper to spare, no doubt the computer could do the job in a fraction of the time!

It would be interesting to speculate the size of the task if the list was typed out in the form of *words* — ie, beginning one, two, three, four, and continuing up to nine hundred and ninety nine thousand, nine hundred and ninety nine one million. A conservative estimate would indicate that this would require 65 million characters — an increase of almost tenfold on the figures given above!

This month the competition is to devise a simple program to convert a number input in digital form into its 'word' equivalent. The routine should be capable of handling numbers of up to nine digits — ie, in the range 1 to 999999999. So if the number input was 123456789, the printed display would read:

ONE HUNDRED AND TWENTY THREE MILLION, FOUR HUNDRED AND FIFTY SIX THOUSAND, SEVEN HUNDRED AND EIGHTY NINE

Listings will be judged on compactness and their ability to produce reasonable English phrases for the range of numbers specified.

Finally, the answer to Srinivara Ramanujan's problem given earlier is 12 cubed, or 10 cubed plus 9 cubed. Both add up to the same total, 1729. Readers who are interested might like to try out the following listing to compute this answer. If the program is left running it will produce a number of other values, greater than 1729, that are also the sums of two cubes in two different ways.

```
 10 A=3
 20 FOR B=1 TO A
 30 T=A*A*A+B*B*B
 40 FOR C=B TO A-1
 50 IF C=B THEN 90
 60 Z=T-C*C*C
 70 D=INT((Z+0.5)^(1/3))
 80 IF D*D*D=Z THEN
    PRINT A;" ";B;" ";C;
    " ";D;" ";T:C=A
 90 NEXT C
100 NEXT B
110 A=A+1:GOTO 20
```

The Answer

This is Gordon Lee's own solution to the April competition see page 26 for results

ANSWER: The cards are (going clockwise from the queen) queen, 9, 9, 4, king, and 5.

Solution: In the listing the values of each of the possible combinations of cards are stored in the array C(6), C(1) is the first card, running clockwise round to C(6) (the Queen). Card 'X' is at C(3).

The three FOR/NEXT loops at lines 30 to 50 represent cards C(10), C(2), and C(5). Once these have been given values card C(4) can be calculated as we are told that C(1) and C(2) sum to the same total as C(4) and C(5). Providing that C(4) is in the range 1 to 13 and is not a 9, the computation proceeds. Line 80 calculates the value of C(3) as we know that the queen and its neighbours total the same as C(3) and the cards either side of it. Again this must be in the range 1 to 13. The next step is to test the total of all six cards and reject all values not equalling 52.

The next information tells us there is only one pair of cards so the routine at lines 120 to 160 checks each card against each of the others to count the number of pairs.

We then test to see if there is a nine present in the current values. Note that at line 210 we only reject sets of values in which there is no nine. The information as given does not exclude the possibility that there is a pair of nines.

If this program is run it prints out six sets of different values. However, the final clue states that if you knew the value of card 'X' you could find the answer. Only in one case (when 'X' is a four,) does this result in a unique answer. For any other value of 'X' there would be more than one possibility.

```
100 DIM C(6)
110 C(6)=12
120 FOR A=1 TO 13:C(1)=A
130 FOR B=1 TO 13:C(2)=B
140 FOR C=1 TO 13:C(5)=C
150 C(4)=C(1)+C(2)-C(5)
160 IF C(4)<1 OR C(4)>13 OR C(4)=9 THEN 320
170 C(3)=(C(5)+C(6)+C(1))-(C(2)+C(4))
180 IF C(3)<1 OR C(3)>13 THEN 320
190 T=C(1)+C(2)+C(3)+C(4)+C(5)+C(6)
200 IF T<>52 THEN 320
210 PAIR=0
220 FOR F=1 TO 5:FOR G=F+1 TO 6
230 IF C(F)=C(G) THEN PAIR=PAIR+1
240 NEXT:NEXT
250 IF PAIR<>1 THEN 320
260 NINE=0
270 FOR F=1 TO 6
280 IF C(F)=9 THEN NINE=NINE+1
290 NEXT
300 IF NINE=0 THEN 320
310 PRINTC(1);C(2);C(3);C(4);C(5);C(6)
320 NEXT:NEXT:NEXT
```

Communication

Write down your problem on the coupon below (make it as brief and legible as possible) together with your name and address and send it to Communication, 49 Alexandra Road, Hounslow, Middlesex TW3 4HP.

Problem . . . .

. . . .

. . . .

. . . .

Name . . . .

Address . . . .

. . . .

. . . .

Adventure Contact

To help puzzled adventurers further, here is an Adventure Helpline — simply fill in the coupon below, stating the name of the adventure, your problem and your name and address, and send it to Dragon User Adventure Helpline, 49 Alexandra Road, Hounslow, Middlesex TW3 4HP.

Adventure . . . .

Problem . . . .

. . . .

Name . . . .

Address . . . .

. . . .

CLASSIFIED ADS

HERE'S MY CLASSIFIED AD.
(please write your copy in capitals on the lines below)

Name . . . .

Address . . . .

. . . . **Tel:** . . . .

35p per word. Addresses must be paid if required, including post code. Please send this form to Classified Dept., Dragon Publications, 49 Alexandra Road, Hounslow, Middlesex TW3 4HP.

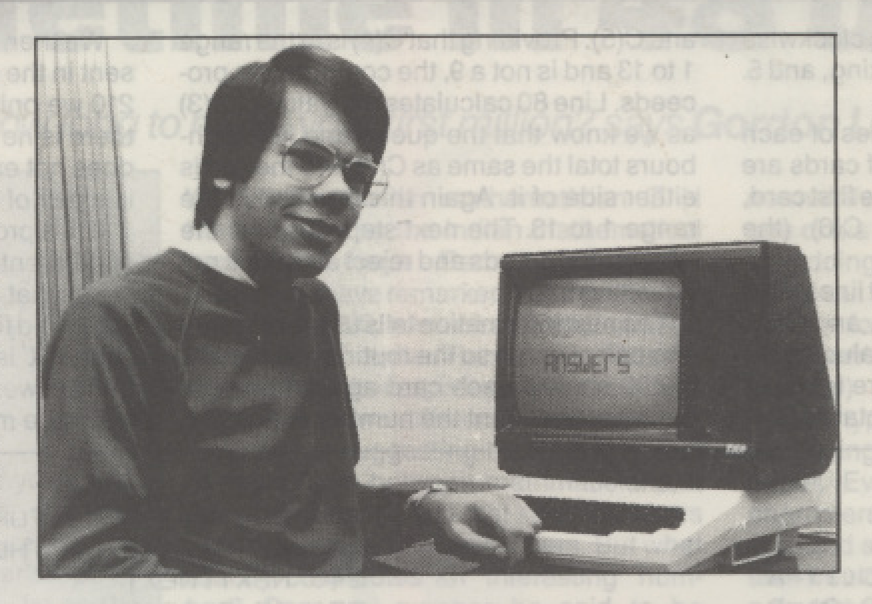
Dragon Answers

If you've got a technical question write to Brian Cadge. Please do not send a SAE as Brian cannot guarantee to answer individual inquiries.

Save to disc

I would like to be able to change the load and save game routines in some of my machine code programs to use the disc rather than the tape. Could you please tell me how to create the equivalent of the SAVE "file", start, end, entry and LOAD commands in machine code?

Ian Burford
79 Hertford Road
Stevenage
SG2 8SE



```
        LDX  166            ; GET CURRENT POINTER
        PSHS X              ; SAVE IT
        LDX  #SAVTXT
        STX  166            ; POINT TO SAVE TEXT
        JSR  54591          ; CALL THE SAVE ROUTINE
        PULS X              ; GET OLD POINTER
        STX  166            ; RESTORE IT
        LDA  #1             ; DRIVE 1
        JSR  [$C010]        ; CALL CLOSAL ROUTINE
        RTS
SAVTXT FCC  /"FILENAME",10000,10500,0/,0
```

THE simplest way of doing this is really a bit of a fudge, but it works. Set up the statement pointer at 166/167 to point to a string containing the required name and save address, length etc. (this must end with a zero byte), then call the Basic SAVE command in rom. The listing below shows how to do this (SAVE's address is 54591, LOAD's is 54439 for DragonDOS 1.0). A call to CLOSAL is made after the SAVE to close the file properly.

You can use a similar technique to simulate the LOAD command from machine code.

Non-stop DREAMing

WHEN loading a tape saved through the Dream editor and assembler package, it starts OK but it never knows when to stop. I have tried all the pokes given in your replies to previous enquiries, but to no avail. All the pokes refer to altering the header; my problem appears to be a lack of EOF marker.

A. Wilson
50 Goodacre
Orton Goldhay
Peterborough
PE2 0LZ

ASSUMING that other files (Basic, binary etc.) load and save without problem you appear to have a problem with Dream that I have not come across before. The assembler uses standard cassette I/O rom calls, and so there should be no special problems (except that it sets a very short header length which previously published pokes overcome).

Have any of our readers had the same problem and found a cure? If so let me know and I'll publish the solution.

Input on the run?

AT the moment I am working on a program which, among other things, contains mathematical functions. However, I cannot find how to input functions while the program is running. I have tried the following, but it does not work:

100 INPUT A$:DEF FNF(X)=A$

Is it possible to input functions while a program is running and how?

Pal Dahle
Dale
1620 Gressvik
Norway

THE above routine does not work because the function definition should not be a string, but rather normal tokenised Basic. Therefore, you cannot enter a new function while the program is running (as you cannot tokenise and store it). The only solution is to have a number of separate subroutines to define all your required functions, and call these as necessary.

Echoes of Tron to printer

I would very much like to have TRON diverted to the printer, with or without screen display. I had hoped that your 'echo screen to printer' listing of some year or two ago would do the trick, but it won't work if DOS is connected, as both use the 359 hook. Is there a way with DOS connected, please?

A. Davis
39 Boxley Drive
West Bridgeford
Notts.

THE routine listed below will do the trick. It echoes screen text to the printer, but only if TRON is activated (by checking the flag at location 175). Also, the hook at 359 is copied and used to exit the routine, so it will work with or without DOS attached.

Run the Basic loader program to install the code, and then load the program you want to trace. The assembly language listing of the code is also given for anyone that's interested.

```
2AC7                *
2AC7                * TRON ECHO TO PRINTER
2AC7                * WORKS WITH DOS TOO
2AC7                *
7FBD  7FBD                  ORG   32701
7FBD                        PUT   58000
7FBD B60167         START   LDA   359
7FC0 BE0168                 LDX   360
7FC3 B77FE4                 STA   TOROM
7FC6 BF7FE5                 STX   TOROM+1
7FC9 867E                   LDA   £126
7FCB B70167                 STA   359
7FCE 8E7FD5                 LDX   £ECHO
7FD1 BF0168                 STX   360
7FD4 39                     RTS
7FD5 3402           ECHO    PSHS  A
7FD7 0D6F                   TST   111
7FD9 2607                   BNE   TROFF
7FDB 0DAF                   TST   175
7FDD 2703                   BEQ   TROFF
7FDF BDBD1A                 JSR   48410
7FE2 3502           TROFF   PULS  A
7FE4 39             TOROM   RTS
7FE5 0000                   FCB   0,0
7FE7
```

```
10 REM TRON ECHO TO PRINTER
20 REM WORKS WITH OR WITHOUT DOS
30 CLEAR 200,32700
40 FOR I=32701 TO 32740
50 READ A$:A=VAL("&H"+A$):C=C+A:POKE I,A:NEXT
60 IF C<>4017 THEN PRINT"DATA ERROR":STOP
70 EXEC 32701:CLS:PRINT"TRON ECHO INSTALLED":END
80 DATA B6,01,67,BE,01,68,B7,7F
90 DATA E4,BF,7F,E5,86,7E,B7,01
100 DATA 67,8E,7F,D5,BF,01,68,39
110 DATA 34,02,0D,6F,26,07,0D,AF
120 DATA 27,03,BD,BD,1A,35,02,39
```